

# Preprocessing

```
In [180]: import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler
from sklearn.preprocessing import LabelEncoder
```

```
In [181]: data=pd.read_csv("AB_NYC_2019.csv")
```

In [182]:

data

Out[182]:

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	
0	2539	Clean & quiet apt home by the park	2787	John	Brooklyn	Kensington	4
1	2595	Skylit Midtown Castle	2845	Jennifer	Manhattan	Midtown	4
2	3647	THE VILLAGE OF HARLEM....NEW YORK !	4632	Elisabeth	Manhattan	Harlem	4
3	3831	Cozy Entire Floor of Brownstone	4869	LisaRoxanne	Brooklyn	Clinton Hill	4
4	5022	Entire Apt: Spacious Studio/Loft by central park	7192	Laura	Manhattan	East Harlem	4
...	...	...	...	...	...	...	...
48890	36484665	Charming one bedroom - newly renovated rowhouse	8232441	Sabrina	Brooklyn	Bedford-Stuyvesant	4
48891	36485057	Affordable room in Bushwick/East Williamsburg	6570630	Marisol	Brooklyn	Bushwick	4
48892	36485431	Sunny Studio at Historical Neighborhood	23492952	Ilgar & Aysel	Manhattan	Harlem	4
48893	36485609	43rd St. Time Square-cozy single bed	30985759	Taz	Manhattan	Hell's Kitchen	4
48894	36487245	Trendy duplex in the very heart of Hell's Kitchen	68119814	Christophe	Manhattan	Hell's Kitchen	4

48895 rows × 16 columns

In [183]: `data.describe()`

Out[183]:

	id	host_id	latitude	longitude	price	minimum_nights	nu
<b>count</b>	4.889500e+04	4.889500e+04	48895.000000	48895.000000	48895.000000	48895.000000	
<b>mean</b>	1.901714e+07	6.762001e+07	40.728949	-73.952170	152.720687	7.029962	
<b>std</b>	1.098311e+07	7.861097e+07	0.054530	0.046157	240.154170	20.510550	
<b>min</b>	2.539000e+03	2.438000e+03	40.499790	-74.244420	0.000000	1.000000	
<b>25%</b>	9.471945e+06	7.822033e+06	40.690100	-73.983070	69.000000	1.000000	
<b>50%</b>	1.967728e+07	3.079382e+07	40.723070	-73.955680	106.000000	3.000000	
<b>75%</b>	2.915218e+07	1.074344e+08	40.763115	-73.936275	175.000000	5.000000	
<b>max</b>	3.648724e+07	2.743213e+08	40.913060	-73.712990	10000.000000	1250.000000	



In [184]: `data.dtypes`

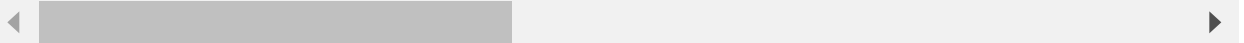
Out[184]:

id	int64
name	object
host_id	int64
host_name	object
neighbourhood_group	object
neighbourhood	object
latitude	float64
longitude	float64
room_type	object
price	int64
minimum_nights	int64
number_of_reviews	int64
last_review	object
reviews_per_month	float64
calculated_host_listings_count	int64
availability_365	int64
dtype:	object

In [185]: `data.head()`

Out[185]:

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	lc
0	2539	Clean & quiet apt home by the park	2787	John	Brooklyn	Kensington	40.64749	-7
1	2595	Skylit Midtown Castle	2845	Jennifer	Manhattan	Midtown	40.75362	-7
2	3647	THE VILLAGE OF HARLEM....NEW YORK !	4632	Elisabeth	Manhattan	Harlem	40.80902	-7
3	3831	Cozy Entire Floor of Brownstone	4869	LisaRoxanne	Brooklyn	Clinton Hill	40.68514	-7
4	5022	Entire Apt: Spacious Studio/Loft by central park	7192	Laura	Manhattan	East Harlem	40.79851	-7



In [186]: `data.shape`

Out[186]: (48895, 16)

In [187]: `data.isnull().sum()`

Out[187]:

id	0
name	16
host_id	0
host_name	21
neighbourhood_group	0
neighbourhood	0
latitude	0
longitude	0
room_type	0
price	0
minimum_nights	0
number_of_reviews	0
last_review	10052
reviews_per_month	10052
calculated_host_listings_count	0
availability_365	0
dtype:	int64

In [188]: `data2=data.dropna(subset=['name', 'host_name'])`

```
In [189]: data2=data2.dropna()
```

```
In [190]: data2.isnull().sum()
```

```
Out[190]: id                0
          name              0
          host_id          0
          host_name        0
          neighbourhood_group 0
          neighbourhood     0
          latitude         0
          longitude        0
          room_type        0
          price            0
          minimum_nights    0
          number_of_reviews 0
          last_review       0
          reviews_per_month 0
          calculated_host_listings_count 0
          availability_365   0
          dtype: int64
```

```
In [191]: data2.shape
```

```
Out[191]: (38821, 16)
```

```
In [192]: data2=data2[(data2.price!= 0) & (data2.minimum_nights!= 0) & (data2.number_of_reviews!= 0)]
          data2.reset_index(drop=True, inplace=True)
```

```
In [193]: mean_availability_365 = data2['availability_365'].mean(skipna=True)
          data2=data2.replace(0,mean_availability_365)
```

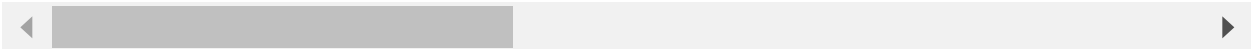
```
In [194]: data2.shape
```

```
Out[194]: (38811, 16)
```

```
In [195]: data2
```

Out[195]:

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latit
0	2539	Clean & quiet apt home by the park	2787	John	Brooklyn	Kensington	40.64
1	2595	Skylit Midtown Castle	2845	Jennifer	Manhattan	Midtown	40.75
2	3831	Cozy Entire Floor of Brownstone	4869	LisaRoxanne	Brooklyn	Clinton Hill	40.68
3	5022	Entire Apt: Spacious Studio/Loft by central park	7192	Laura	Manhattan	East Harlem	40.79
4	5099	Large Cozy 1 BR Apartment In Midtown East	7322	Chris	Manhattan	Murray Hill	40.74
...	...	...	...	...	...	...	...
38806	36425863	Lovely Privet Bedroom with Privet Restroom	83554966	Rusaa	Manhattan	Upper East Side	40.78
38807	36427429	No.2 with queen size bed	257683179	H Ai	Queens	Flushing	40.75
38808	36438336	Seas The Moment	211644523	Ben	Staten Island	Great Kills	40.54
38809	36442252	1B-1B apartment near by Metro	273841667	Blaine	Bronx	Mott Haven	40.80
38810	36455809	Cozy Private Room in Bushwick, Brooklyn	74162901	Christine	Brooklyn	Bushwick	40.69
38811 rows × 16 columns							



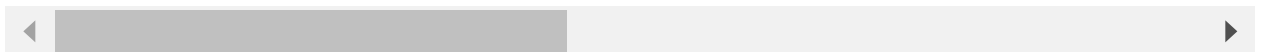
```
In [196]: data3=data2
del data3['name']
del data3['host_name']
```

In [197]: data3

Out[197]:

	id	host_id	neighbourhood_group	neighbourhood	latitude	longitude	room_type
0	2539	2787	Brooklyn	Kensington	40.64749	-73.97237	Private room
1	2595	2845	Manhattan	Midtown	40.75362	-73.98377	Entire home/ap
2	3831	4869	Brooklyn	Clinton Hill	40.68514	-73.95976	Entire home/ap
3	5022	7192	Manhattan	East Harlem	40.79851	-73.94399	Entire home/ap
4	5099	7322	Manhattan	Murray Hill	40.74767	-73.97500	Entire home/ap
...	...	...	...	...	...	...	...
38806	36425863	83554966	Manhattan	Upper East Side	40.78099	-73.95366	Private room
38807	36427429	257683179	Queens	Flushing	40.75104	-73.81459	Private room
38808	36438336	211644523	Staten Island	Great Kills	40.54179	-74.14275	Private room
38809	36442252	273841667	Bronx	Mott Haven	40.80787	-73.92400	Entire home/ap
38810	36455809	74162901	Brooklyn	Bushwick	40.69805	-73.92801	Private room

38811 rows × 14 columns



In [198]: data3['neighbourhood\_group'].value\_counts(normalize=True)

Out[198]: Manhattan 0.428255  
 Brooklyn 0.423334  
 Queens 0.117802  
 Bronx 0.022519  
 Staten Island 0.008090  
 Name: neighbourhood\_group, dtype: float64

In [199]: data3['room\_type'].value\_counts(normalize=True)

Out[199]: Entire home/apt 0.523563  
 Private room 0.454691  
 Shared room 0.021746  
 Name: room\_type, dtype: float64

```
In [200]: data3['price'].describe()
```

```
Out[200]: count      38811.000000
mean         142.369199
std          197.006883
min           10.000000
25%           69.000000
50%          101.000000
75%          170.000000
max          10000.000000
Name: price, dtype: float64
```

## Discretization on numerical variables 'price' and 'availability\_365'

```
In [201]: data3.dtypes
```

```
Out[201]: id                int64
host_id                  int64
neighbourhood_group      object
neighbourhood            object
latitude                 float64
longitude                float64
room_type                object
price                    int64
minimum_nights            int64
number_of_reviews         int64
last_review              object
reviews_per_month         float64
calculated_host_listings_count  int64
availability_365          float64
dtype: object
```

```
In [202]: Q1=data3['price'].quantile(0.25)
Q2=data3['price'].quantile(0.50)
Q3=data3['price'].quantile(0.75)
IQR=Q3-Q1
uw=Q3+1.5*IQR
lw=Q1-1.5*IQR

column='price'
data3[column] = np.where((data3[column]<lw) , 'lower_outlier',
                        np.where((data3[column] >=lw) & (data3[column] <=Q1),
                        np.where((data3[column]>Q1) & (data3[column] <=Q2),
                        np.where((data3[column]>Q2) & (data3[column] <=Q3),
                        np.where((data3[column]>Q3) & (data3[column] <=Q3+1.5*IQR),
                        'upper_outlier'))))
```



```
In [203]: data3['price'].value_counts()
```

```
Out[203]: low                9974
          high_median        9864
          low_median         9440
          high               7456
          upper_outlier      2077
          Name: price, dtype: int64
```

```
In [204]: data3['availability_365'].describe()
```

```
Out[204]: count    38811.000000
          mean       152.394061
          std        105.178907
          min         1.000000
          25%         89.000000
          50%        114.881631
          75%        229.000000
          max        365.000000
          Name: availability_365, dtype: float64
```

```
In [205]: data3['availability_365'] = np.where((data3['availability_365'] >=0) & (data3['av
          np.where((data3['availability_365'] >90) & (data3[
          np.where((data3['availability_365'] >180) & (data3[
          np.where((data3['availability_365'] >270) & (data3[
          'no_booking')
```

```
In [206]: data3['availability_365'].value_counts()
```

```
Out[206]: mostly_booked      17199
          Fully_booked        9968
          rarely_booked       6169
          frequently_booked    3960
          no_booking          1515
          Name: availability_365, dtype: int64
```

```
In [207]: del data3['id']
          del data3['host_id']
          del data3['neighbourhood']
          del data3['latitude']
          del data3['longitude']
          del data3['last_review']
```

In [208]: data3

Out[208]:

	neighbourhood_group	room_type	price	minimum_nights	number_of_reviews	review
0	Brooklyn	Private room	high_median	1	9	
1	Manhattan	Entire home/apt	high	1	45	
2	Brooklyn	Entire home/apt	low_median	1	270	
3	Manhattan	Entire home/apt	low_median	10	9	
4	Manhattan	Entire home/apt	high	3	74	
...	...	...	...	...	...	...
38806	Manhattan	Private room	high_median	1	1	
38807	Queens	Private room	low	1	1	
38808	Staten Island	Private room	high	1	1	
38809	Bronx	Entire home/apt	low_median	1	2	
38810	Brooklyn	Private room	low	1	1	

38811 rows × 8 columns

```

In [68]: def data_splitter(data3):
          num_col=[]
          cat_col=[]
          for col in data.columns:
              if ((data[col].dtype=='int64') or (data[col].dtype=='float64')):
                  num_col.append(col)
              else:
                  cat_col.append(col)
          num_data=data[num_col]
          cat_data=data[cat_col]

          return num_data,cat_data

```

In [70]: num, cat=data\_splitter(data3)

In [71]: num

Out[71]:

	minimum_nights	number_of_reviews	reviews_per_month	calculated_host_listings_count
0	1	9	0.21	6
1	1	45	0.38	2
2	1	270	4.64	1
3	10	9	0.10	1
4	3	74	0.59	1
...	...	...	...	...
38806	1	1	1.00	1
38807	1	1	1.00	6
38808	1	1	1.00	1
38809	1	2	2.00	1
38810	1	1	1.00	1

38811 rows × 4 columns

## Normalization

In [72]: `from sklearn.preprocessing import StandardScaler`  
`scaler=StandardScaler()`  
`num_norm=scaler.fit_transform(num).round(2)`

In [73]: num\_norm

Out[73]: array([[ -0.28, -0.42, -0.69, 0.03],  
 [-0.28, 0.33, -0.59, -0.12],  
 [-0.28, 5. , 1.94, -0.16],  
 ...,  
 [-0.28, -0.59, -0.22, -0.16],  
 [-0.28, -0.57, 0.37, -0.16],  
 [-0.28, -0.59, -0.22, -0.16]])

In [74]: `df_num_norm=pd.DataFrame(num_norm, columns=num.columns)`

In [75]: `df_num_norm`

Out[75]:

	minimum_nights	number_of_reviews	reviews_per_month	calculated_host_listings_count
0	-0.28	-0.42	-0.69	0.03
1	-0.28	0.33	-0.59	-0.12
2	-0.28	5.00	1.94	-0.16
3	0.24	-0.42	-0.76	-0.16
4	-0.16	0.93	-0.47	-0.16
...	...	...	...	...
38806	-0.28	-0.59	-0.22	-0.16
38807	-0.28	-0.59	-0.22	0.03
38808	-0.28	-0.59	-0.22	-0.16
38809	-0.28	-0.57	0.37	-0.16
38810	-0.28	-0.59	-0.22	-0.16

38811 rows × 4 columns

In [76]: `data4 = pd.concat([df_num_norm, cat], axis=1)`

In [78]: `data4.dtypes`

Out[78]:

minimum_nights	float64
number_of_reviews	float64
reviews_per_month	float64
calculated_host_listings_count	float64
neighbourhood_group	object
room_type	object
price	object
availability_365	object
dtype:	object

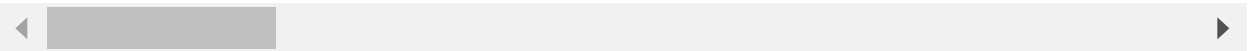
**Apply one-hot encoding to all the categorical attributes  
(Preserve the class attribute)**

```
In [81]: df_dummies = pd.get_dummies(data4)
df_dummies.head()
```

```
Out[81]:
```

	minimum_nights	number_of_reviews	reviews_per_month	calculated_host_listings_count	neighboc
0	-0.28	-0.42	-0.69	0.03	
1	-0.28	0.33	-0.59	-0.12	
2	-0.28	5.00	1.94	-0.16	
3	0.24	-0.42	-0.76	-0.16	
4	-0.16	0.93	-0.47	-0.16	

5 rows × 22 columns



## Train test

```
In [ ]: # We will use the data frame where we had created dummy variables
y = df_dummies['y'].values
X = df_dummies.drop(columns = ['y'])
```

```
In [86]: from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_s
```

```
-----
NameError                                Traceback (most recent call last)
Input In [86], in <cell line: 2>()
      1 from sklearn.model_selection import train_test_split
----> 2 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3
, random_state=101)
```

**NameError**: name 'X' is not defined