Open in Colab

PARCIAL 1

MINE-4101: Applied Data Science
Universidad de los Andes

Last update: September, 2022

Dataset: Información de la inmobiliaria los Andes Source: UCI Machine Learning Repository

Código de Honor. Al entregar la solución de este parcial, yo, Fabián Camilo Castellanos Pinto con código 202226029 me comprometo a no conversar durante el desarrollo de este examen con ninguna persona que no sea el profesor del curso, sobre aspectos relacionados con el parcial; tampoco utilizaré algún medio de comunicación por voz, texto o intercambio de archivos, para consultar o compartir con otros, información sobre el tema del parcial. Soy consciente y acepto las consecuencias que acarreará para mi desempeño académico cometer fraude en este parcial

Task: Utilizar el mejor modelo obtenido para estimar la popularidad de los inmuebles próximos a publicarse.

Diccionario de datos:

1. **id**: Identificador del inmueble
2. **neighbourhood group**: Localidad o distrito en el que se encuentra el inmueble
3. **neighbourhood**: Barrio en el que se encuentra el inmueble
4. **lat**, **long**: Geolocalización del inmueble
5. **country**: Pais en el que se encuentra el inmueble
6. **instant bookable**: Indicador de si es posible realizar reserva directamente en la plataforma
7. **cancellation_policy**: Política de cancelación de la reserva
8. **room type**: Tipo de inmueble
9. **construction year**: Año de construcción del inmueble

10. **price**: Precio por noche del inmueble
11. **service fee**: Costo del servicio el cual debe ser cancelado al dejar el inmueble
12. **minimum nights**: Cantidad mínima de noches que el inmueble puede ser reservado
13. **availability 365**: Disponibilidad total en días durante el último año
14. **number of reviews**: Total de comentarios del inmueble
15. **review rate number**: Calificación promedio dada al inmueble

```
!pip install --upgrade pandas-profiling
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: pandas-profiling in /usr/local/lib/python3.7/dist-packages (3.3.0)
Requirement already satisfied: missingno<0.6,>=0.4.2 in /usr/local/lib/python3.7/dist-packages (from pandas-profiling) (0.5.1)
Requirement already satisfied: phik<0.13,>=0.11.1 in /usr/local/lib/python3.7/dist-packages (from pandas-profiling) (0.12.2)
Requirement already satisfied: scipy<1.10,>=1.4.1 in /usr/local/lib/python3.7/dist-packages (from pandas-profiling) (1.7.3)
Requirement already satisfied: tangled-up-in-unicode==0.2.0 in /usr/local/lib/python3.7/dist-packages (from pandas-profiling) (
Requirement already satisfied: seaborn<0.12,>=0.10.1 in /usr/local/lib/python3.7/dist-packages (from pandas-profiling) (0.11.2)
Requirement already satisfied: numpy<1.24,>=1.16.0 in /usr/local/lib/python3.7/dist-packages (from pandas-profiling) (1.21.6)
Requirement already satisfied: requests<2.29,>=2.24.0 in /usr/local/lib/python3.7/dist-packages (from pandas-profiling) (2.28.1
Requirement already satisfied: htmlmin==0.1.12 in /usr/local/lib/python3.7/dist-packages (from pandas-profiling) (0.1.12)
Requirement already satisfied: tqdm<4.65,>=4.48.2 in /usr/local/lib/python3.7/dist-packages (from pandas-profiling) (4.64.1)
Requirement already satisfied: joblib~=1.1.0 in /usr/local/lib/python3.7/dist-packages (from pandas-profiling) (1.1.0)
Requirement already satisfied: jinja2<3.2,>=2.11.1 in /usr/local/lib/python3.7/dist-packages (from pandas-profiling) (2.11.3)
Requirement already satisfied: visions[type_image_path]==0.7.5 in /usr/local/lib/python3.7/dist-packages (from pandas-profiling
Requirement already satisfied: pandas!=1.4.0,<1.5,>1.1 in /usr/local/lib/python3.7/dist-packages (from pandas-profiling) (1.3.5
Requirement already satisfied: statsmodels<0.14,>=0.13.2 in /usr/local/lib/python3.7/dist-packages (from pandas-profiling) (0.1
Requirement already satisfied: PyYAML<6.1,>=5.0.0 in /usr/local/lib/python3.7/dist-packages (from pandas-profiling) (6.0)
Requirement already satisfied: multimethod<1.9,>=1.4 in /usr/local/lib/python3.7/dist-packages (from pandas-profiling) (1.8)
Requirement already satisfied: pydantic<1.10,>=1.8.1 in /usr/local/lib/python3.7/dist-packages (from pandas-profiling) (1.9.2)
Requirement already satisfied: matplotlib<3.6,>=3.2 in /usr/local/lib/python3.7/dist-packages (from pandas-profiling) (3.5.3)
Requirement already satisfied: networkx>=2.4 in /usr/local/lib/python3.7/dist-packages (from visions[type_image_path]==0.7.5->p
Requirement already satisfied: attrs>=19.3.0 in /usr/local/lib/python3.7/dist-packages (from visions[type_image_path]==0.7.5->p
Requirement already satisfied: Pillow in /usr/local/lib/python3.7/dist-packages (from visions[type_image_path]==0.7.5->pandas-p
Requirement already satisfied: imagehash in /usr/local/lib/python3.7/dist-packages (from visions[type_image_path]==0.7.5->panda
Requirement already satisfied: MarkupSafe>=0.23 in /usr/local/lib/python3.7/dist-packages (from jinja2<3.2,>=2.11.1->pandas-pro
Requirement already satisfied: pyparsing>=2.2.1 in /usr/local/lib/python3.7/dist-packages (from matplotlib<3.6,>=3.2->pandas-pr
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.7/dist-packages (from matplotlib<3.6,>=3.2->pandas-p
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.7/dist-packages (from matplotlib<3.6,>=3.2->panda
```

```
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.7/dist-packages (from matplotlib<3.6,>=3.2->pandas-pro
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.7/dist-packages (from matplotlib<3.6,>=3.2->pandas-profil
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.7/dist-packages (from matplotlib<3.6,>=3.2->pandas-p
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.7/dist-packages (from kiwisolver>=1.0.1->matplotlib<
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-packages (from pandas!=1.4.0,<1.5,>1.1->pandas-pro
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages (from python-dateutil>=2.7->matplotlib<3.6,>=
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.7/dist-packages (from requests<2.29,>=2.24.0->pa
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.7/dist-packages (from requests<2.29,>=2.24.0->pandas-prof
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packages (from requests<2.29,>=2.24.0->panda
Requirement already satisfied: charset-normalizer<3,>=2 in /usr/local/lib/python3.7/dist-packages (from requests<2.29,>=2.24.0-
Requirement already satisfied: patsy>=0.5.2 in /usr/local/lib/python3.7/dist-packages (from statsmodels<0.14,>=0.13.2->pandas-p
Requirement already satisfied: PyWavelets in /usr/local/lib/python3.7/dist-packages (from imagehash->visions[type_image_path]==
```

```
!pip install markupsafe==2.0.1
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: markupsafe==2.0.1 in /usr/local/lib/python3.7/dist-packages (2.0.1)
```

```
!pip install --upgrade matplotlib
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: matplotlib in /usr/local/lib/python3.7/dist-packages (3.5.3)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.7/dist-packages (from matplotlib) (21.3)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.7/dist-packages (from matplotlib) (1.21.6)
Requirement already satisfied: pyparsing>=2.2.1 in /usr/local/lib/python3.7/dist-packages (from matplotlib) (3.0.9)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.7/dist-packages (from matplotlib) (4.37.3)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.7/dist-packages (from matplotlib) (2.8.2)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.7/dist-packages (from matplotlib) (7.1.2)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.7/dist-packages (from matplotlib) (0.11.0)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.7/dist-packages (from matplotlib) (1.4.4)
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.7/dist-packages (from kiwisolver>=1.0.1->matplotlib)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages (from python-dateutil>=2.7->matplotlib) (1.15
```

```
import numpy as np
import pandas as pd
```

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, mean_absolute_error

import matplotlib.pyplot as plt
import seaborn as sns

from pandas_profiling import ProfileReport
import pylev
```

Leer y explorar las fuentes de datos

```
nuevo_df = pd.read_csv('losalpes_new.csv', sep = ',')
historico_df = pd.read_csv('losalpes_history.csv', sep = ',')
```

```
historico_df.shape
```

```
(102083, 16)
```

```
historico_df.dtypes
```

```
id                       int64
neighbourhood group     object
neighbourhood           object
lat                    float64
long                   float64
country                 object
instant_bookable        object
cancellation_policy     object
room type               object
construction year      float64
price                   object
service fee             object
```

```
minimum nights         float64
availability 365       float64
number of reviews      float64
review rate number     float64
dtype: object
```

```
historico_df
```

	id	neighbourhood group	neighbourhood	lat	long	country	instant_bookable	cancellation_policy	room type	c
0	48540006	Manhattan	Hell's Kitchen	40.76212	-73.98820	United States	True	strict	Entire home/apt	

```
historico_df.head()
```

	id	neighbourhood group	neighbourhood	lat	long	country	instant_bookable	cancellation_policy	room type	constr
0	48540006	Manhattan	Hell's Kitchen	40.76212	-73.98820	United States	True	strict	Entire home/apt	
1	35079903	Manhattan	Midtown	40.74623	-73.98499	United States	True	flexible	Hotel room	
2	50681273	Manhattan	Upper West Side	40.78859	-73.97568	United States	False	flexible	Private room	
3	13039267	Manhattan	Financial District	40.70817	-74.00511	United States	False	strict	Private room	
4	8998640	Manhattan	Lower East Side	40.72130	-73.98900	United States	True	moderate	Entire home/apt	

```
nuevo_df.shape
```

```
(513, 14)
```

```
nuevo_df.dtypes
```

```
id                          int64
neighbourhood group        object
```

```
neighbourhood           object
lat                    float64
long                   float64
country                 object
instant_bookable        object
cancellation_policy     object
room type               object
construction year      float64
price                   object
service fee             object
minimum nights         float64
availability 365       float64
dtype: object
```

```
nuevo_df
```

	id	neighbourhood group	neighbourhood	lat	long	country	instant_bookable	cancellation_policy	room type	cons
0	27883434	Queens	Ozone Park	40.68432	-73.85862	United States	False	moderate	Private room	
1	55448727	Manhattan	Civic Center	40.71317	-74.00654	United States	False	moderate	Entire home/apt	
2	56858749	Queens	East Elmhurst	40.76441	-73.88943	NaN	True	flexible	Private room	
						United			Private	

```
nuevo_df.head()
```

	id	neighbourhood group	neighbourhood	lat	long	country	instant_bookable	cancellation_policy	room type	constr
0	27883434	Queens	Ozone Park	40.68432	-73.85862	United States	False	moderate	Private room	
1	55448727	Manhattan	Civic Center	40.71317	-74.00654	United States	False	moderate	Entire home/apt	
2	56858749	Queens	East Elmhurst	40.76441	-73.88943	NaN	True	flexible	Private room	
3	39029953	Manhattan	Gramercy	40.73442	-73.98383	United States	True	strict	Private room	
4	5567200	Manhattan	Upper West Side	40.79660	-73.97154	United States	True	strict	Entire home/apt	

Verificando el profile de los datos

```
profile_hist = ProfileReport(historico_df)
```

```
profile_hist.to_notebook_iframe()
```

Summarize dataset: 100% 80/80 [00:35<00:00, 3.92it/s, Completed]

Generate report structure: 100% 1/1 [00:07<00:00, 7.31s/it]

Render HTML: 100% 1/1 [00:02<00:00, 2.59s/it]

neighbourhood
Categorical

HIGH CARDINALITY

Distinct	224
Distinct (%)	0.2%
Missing	607

```
profile_nuevo = ProfileReport(nuevo_df)
```

```
profile_nuevo.to_notebook_iframe()
```

Summarize dataset: 100% 65/65 [00:10<00:00, 3.63it/s, Completed]

Generate report structure: 100% 1/1 [00:07<00:00, 7.00s/it]

Render HTML: 100% 1/1 [00:01<00:00, 1.96s/it]

t	456	11.0%
o	412	9.9%
M	224	5.4%
h	224	5.4%
B	213	5.1%
r	213	5.1%
l	203	4.9%
y	199	4.8%
Other values (10)	592	14.2%

Most occurring categories

Value	Count	Frequency (%)
Lowercase Letter	3636	87.5%

Exploración adicional de la información

Reviso los valores únicos de las columnas de los 2 datasets

```
historico_df["neighbourhood group"].value_counts()
```

```
Manhattan        43384
Brooklyn         41437
Queens           13015
Bronx             2666
Staten Island      943
Broolkyn             7
Quens                4
Manhatan             4
Manattan             1
brookln              1
manhatan             1
Name: neighbourhood group, dtype: int64
```

```
nuevo_df["neighbourhood group"].value_counts()
```

```
Manhattan        224
Brooklyn         199
Queens            71
Bronx             14
Staten Island      4
Name: neighbourhood group, dtype: int64
```

```
historico_df["neighbourhood"].value_counts()
```

```
Bedford-Stuyvesant        7857
Williamsburg              7720
Harlem                    5400
Bushwick                  4930
Hell's Kitchen            3927
                          ...
Lighthouse Hill              3
Gerritsen Beach              3
Glen Oaks                    2
Fort Wadsworth               2
Chelsea, Staten Island       1
Name: neighbourhood, Length: 224, dtype: int64
```

```
nuevo_df["neighbourhood"].value_counts()
```

```
Bedford-Stuyvesant    41
Williamsburg          38
Harlem                29
Hell's Kitchen        21
Bushwick              21
                      ..
Kensington             1
Huguenot               1
Bensonhurst            1
Carroll Gardens        1
Van Nest               1
Name: neighbourhood, Length: 94, dtype: int64
```

```
historico_df["country"].value_counts()
```

```
United States               100957
United States of America        10
Name: country, dtype: int64
```

```
nuevo_df["country"].value_counts()
```

```
United States    507
Name: country, dtype: int64
```

```
historico_df["service fee"].value_counts()
```

```
$ 216       519
$ 177       517
$ 41        517
$ 81        514
$ 57        506
           ...
$ 58        376
$ 10        262
$ 240       247
```

```
$ 122000      1
$ -193        1
Name: service fee, Length: 233, dtype: int64
```

```
nuevo_df["service fee"].value_counts()
```

```
$ 139    7
$ 214    6
$ 164    6
$ 15     5
$ 238    5
        ..
$ 24     1
$ 121    1
$ 91     1
$ 143    1
$ 22     1
Name: service fee, Length: 203, dtype: int64
```

```
historico_df["room type"].value_counts()
```

```
Entire home/apt    52920
Private room       45763
Shared room         2166
Hotel room           113
Name: room type, dtype: int64
```

```
nuevo_df["room type"].value_counts()
```

```
Entire home/apt    263
Private room       237
Shared room         10
Hotel room           2
Name: room type, dtype: int64
```

```
historico_df["construction year"].value_counts()
```

```
2014.0    5162
2008.0    5149
2006.0    5146
2019.0    5121
2020.0    5089
2010.0    5081
2009.0    5073
2005.0    5051
2022.0    5050
2012.0    5041
2003.0    5038
2007.0    5035
2015.0    5003
2011.0    4979
2017.0    4976
2018.0    4971
2021.0    4967
2016.0    4940
2013.0    4937
2004.0    4937
1022.0       2
1020.0       1
Name: construction year, dtype: int64
```

```
nuevo_df["construction year"].value_counts()
```

```
2015.0    38
2004.0    33
2022.0    31
2012.0    31
2014.0    30
2003.0    29
2016.0    27
2018.0    27
2019.0    26
2006.0    25
2009.0    25
2011.0    25
2013.0    24
2020.0    24
```

```
2007.0    23
2017.0    23
2008.0    20
2021.0    19
2010.0    18
2005.0    14
Name: construction year, dtype: int64
```

Verifico los duplicados

```
nuevo_dup_df = nuevo_df[nuevo_df.duplicated()]
```

```
def tiene_duplicados(reg):
  res = 'no hay duplicado'
  if(reg > 0):
    res = 'tienen {} duplicado(s)'.format(reg)
  return res

print("Los datos históricos {}".format(tiene_duplicados(historico_df.shape[0])))
print("Los datos nuevos {}".format(tiene_duplicados(nuevo_df.shape[0])))
```

```
Los datos históricos tienen 102083 duplicado(s)
Los datos nuevos tienen 513 duplicado(s)
```

Borro los datos históricos duplicados

```
historico_df = historico_df.drop_duplicates()
```

```
historico_df.shape
```

```
(101547, 16)
```

LIMPIEZA DE DATOS

```
# Reemplazar simbolos

def arregla_dinero(x):
    x = str(x).replace('$', '').replace(",", ".").lstrip()
    return float(x)

historico_df['service fee'] = historico_df['service fee'].apply(arregla_dinero)
historico_df['price'] = historico_df['price'].apply(arregla_dinero)
nuevo_df['service fee'] = nuevo_df['service fee'].apply(arregla_dinero)
nuevo_df['service fee'] = nuevo_df['service fee'].apply(arregla_dinero)
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:7: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-ve
  import sys
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:8: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-ve
```

```
historico_df.dtypes
```

```
id                      int64
neighbourhood group    object
neighbourhood          object
lat                   float64
long                  float64
country                object
instant_bookable       object
cancellation_policy    object
room type              object
```

```
construction year      float64
price                  float64
service fee            float64
minimum nights         float64
availability 365       float64
number of reviews      float64
review rate number     float64
dtype: object
```

```
nuevo_df.dtypes
```

```
id                      int64
neighbourhood group    object
neighbourhood          object
lat                   float64
long                  float64
country                object
instant_bookable       object
cancellation_policy    object
room type              object
construction year     float64
price                  object
service fee           float64
minimum nights        float64
availability 365      float64
dtype: object
```

Ajustando los valores de ortografía

```
neigh_group_list = ["Manhattan","Brooklyn","Queens","Bronx","Staten Island"]
```

```
def arregla_grupo(d):
  distance = 10
  index = -1
  res = d
  for n in neigh_group_list:
```

```
        try:
          new_distance = pylev.levenshtein(n, d)
          if(new_distance < distance):
            distance = new_distance
            index = neigh_group_list.index(n)
        except:
          pass

      if(index > -1):
        res = neigh_group_list[index]

      return res

historico_df["neighbourhood group"] = historico_df["neighbourhood group"].apply(arregla_grupo)
nuevo_df["neighbourhood group"] = nuevo_df["neighbourhood group"].apply(arregla_grupo)
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:22: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-ve
```

```
historico_df
```

	id	neighbourhood group	neighbourhood	lat	long	country	instant_bookable	cancellation_policy	room type	c
0	48540006	Manhattan	Hell's Kitchen	40.76212	-73.98820	United States	True	strict	Entire home/apt	
1	35079903	Manhattan	Midtown	40.74623	-73.98499	United States	True	flexible	Hotel room	
2	50681273	Manhattan	Upper West Side	40.78859	-73.97568	United States	False	flexible	Private room	
3	13039267	Manhattan	Financial District	40.70817	-74.00511	United States	False	strict	Private room	
4	8998640	Manhattan	Lower East Side	40.72130	-73.98900	United States	True	moderate	Entire home/apt	
...	...	...	...	...	...	...	...	...	...	...
102078	4462048	Brooklyn	Prospect Heights	40.68137	-73.97081	United States	False	flexible	Entire home/apt	
102079	31315978	Brooklyn	Williamsburg	40.70951	-73.96443	United States	False	strict	Private room	

```
historico_df["neighbourhood group"].value_counts()
```

```
Manhattan        43159
Brooklyn         41236
Queens           12952
Bronx             2649
Staten Island      937
Name: neighbourhood group, dtype: int64
```

101547 rows × 16 columns

```
nuevo_df["neighbourhood group"].value_counts()
```

```
Manhattan        224
Brooklyn         199
Queens            71
```

```
Bronx             14
Staten Island      4
Name: neighbourhood group, dtype: int64
```

Ajustando los valores de los años de construcción

```
historico_df = historico_df.replace({ 'construction year': {1022: 2022, 1020: 2020} })
```

```
historico_df["construction year"].value_counts()
```

```
2014.0    5162
2008.0    5149
2006.0    5146
2019.0    5121
2020.0    5090
2010.0    5081
2009.0    5073
2022.0    5052
2005.0    5051
2012.0    5041
2003.0    5038
2007.0    5035
2015.0    5003
2011.0    4979
2017.0    4976
2018.0    4971
2021.0    4967
2016.0    4940
2013.0    4937
2004.0    4937
Name: construction year, dtype: int64
```

La columna de país es constante, no aporta al modelo, las borro de los datasets

```
del historico_df["country"]
```

```
del nuevo_df["country"]
```

La columna del identificador no aporta a la regresión

```
del historico_df["id"]
```

```
historico_df.dtypes
```

```
neighbourhood group     object
neighbourhood           object
lat                    float64
long                   float64
instant_bookable        object
cancellation_policy     object
room type               object
construction year      float64
price                  float64
service fee            float64
minimum nights         float64
availability 365       float64
number of reviews      float64
review rate number     float64
dtype: object
```

```
nuevo_df.dtypes
```

```
id                      int64
neighbourhood group     object
neighbourhood           object
lat                    float64
long                   float64
instant_bookable        object
cancellation_policy     object
room type               object
construction year      float64
```

```
price                  object
service fee           float64
minimum nights        float64
availability 365      float64
dtype: object
```

Análisis del objetivo

```
historico_df['number of reviews'].describe(percentiles = [.25, .5, .75, .95, .99])
```

```
count    100783.000000
mean         27.453102
std          49.534852
min           0.000000
25%           1.000000
50%           7.000000
75%          30.000000
95%         125.000000
99%         232.000000
max        1024.000000
Name: number of reviews, dtype: float64
```

```
print("El promedio de la variable objetivo es: {}".format(historico_df['number of reviews'].mean()))
```

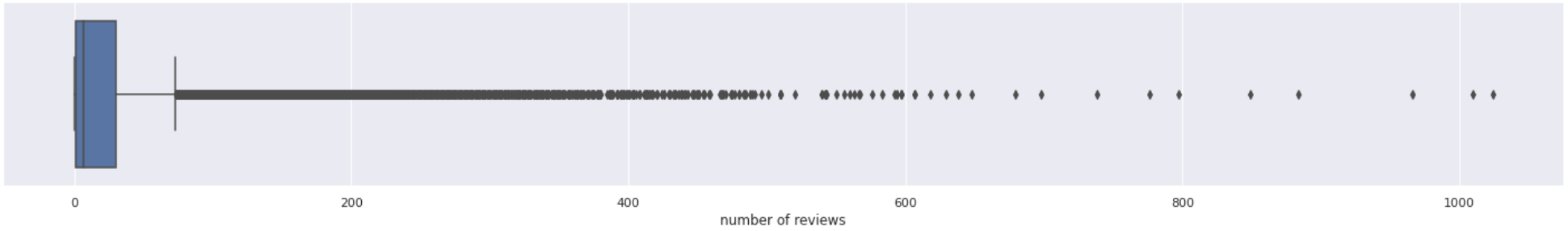
```
El promedio de la variable objetivo es: 27.453102209698063
```

```
print("La desviación estándar de la variable objetivo es: {}".format(historico_df['number of reviews'].std()))
```

```
La desviación estándar de la variable objetivo es: 49.534851919815466
```

```
#sns.set(rc={'figure.figsize':(30,12)})
plt.figure(figsize = (25, 3))
sns.boxplot(historico_df['number of reviews'])
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg:
  FutureWarning
```




```
historico_sel_df = historico_df.loc[historico_df['number of reviews'] <= historico_df['number of reviews'].quantile(.95)]
```

Entrenamiento del modelo

Seleccionando la variables

```
variables = ['lat','long','construction year','price','service fee','minimum nights','availability 365','review rate number']
```

```
historico_sel_df[variables + ['number of reviews']].corr()
```

	lat	long	construction year	price	service fee	minimum nights	availability 365	review rate number	number of reviews
lat	1.000000	0.077701	0.006747	0.002846	-0.002138	0.015951	-0.005195	-0.003304	-0.035010
long	0.077701	1.000000	0.002371	-0.001504	0.002394	-0.038747	0.056624	0.015524	0.080387
construction year	0.006747	0.002371	1.000000	-0.005036	-0.005378	-0.000536	-0.007050	0.004502	0.004045
price	0.002846	-0.001504	-0.005036	1.000000	0.006024	-0.002172	0.002491	0.007652	-0.001673
minimum	0.015951	0.038747	0.000536	0.002172	0.000031	1.000000	0.067940	0.003682	0.057968

```
X = historico_sel_df[variables]
```

```
X
```

```
Y = historico_sel_df['number of reviews']
```

```
0     40.76212  -73.98820  2011.0  851.00  170.0  4.0  199.0  1.0
```

```
Y
```

```
0         112.0
1          13.0
2           1.0
3           5.0
4           9.0
          ...
102078     27.0
102079      7.0
102080     12.0
102081      0.0
102082      1.0
Name: number of reviews, Length: 95803, dtype: float64
```

```
102081  40.78012  -73.98439  2007.0  316.00  63.0  1.0  302.0  NaN
```

Particionando los datos

95803 rows × 8 columns

```
X_entrena, X_prueba, Y_entrena, Y_prueba = train_test_split(X, Y, test_size = 0.2, random_state = 1)
```

Displaying Data

Transformating Data

Colab paid products - Cancel contracts here

✓ 0s completed at 8:47 PM