Prueba de Conocimiento Datos no Estructurados – Eric Acosta

Negocio: *Prueba de Conocimiento Datos no Estructurados*

- Descripción:

La estimación de los precios de la vivienda es esencial tanto para los propietarios como para los inversores, ya que ambos necesitan entender el valor de este activo inmobiliario. Para muchas personas, comprar una propiedad es una de las decisiones y compras más importantes en la vida.

Además de la asequibilidad de una vivienda, otros factores, como la conveniencia del lugar y las perspectivas de inversión a largo plazo, también afectan el proceso de toma de decisiones. El mercado inmobiliario está expuesto a muchas fluctuaciones en los precios debido a las correlaciones existentes con muchas variables, algunas de las cuales no se pueden controlar o incluso pueden ser desconocidas. Los precios de las viviendas pueden aumentar rápidamente (o en algunos casos, también bajan muy rápido)

Negocio: *Prueba de Conocimiento Datos no Estructurados*

- Desarrollo – Primeros pasos:

Desarrollo

Importar librerías

In [1]:
```
#Procesamiento
import pandas as pd
import numpy as np
import datetime as dt
import missingno as msno

#Visualización
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.graph_objects as go
```

Importar CSV

In [2]:
```
#Se suben como archivos temporales para evitar cargarlos a la web y que la información quede pública
test_precios = pd.read_csv("/content/test_precios_vivienda.csv")
train_precios = pd.read_csv("/content/train_precios_vivienda.csv")
```

El primer paso es identificar las librerías que se van a utilizar en el proyecto, en este caso es fundamental Pandas para el manejo de los DataFrame e igual que Matplotlib y Seaborn para poder realizar graficas.

Luego de eso cargué los archivos CSV para poder realizar el desarrollo de la prueba

Negocio: *Prueba de Conocimiento Datos no Estructurados*

- Desarrollo – Limpieza de datos:

Limpieza

Realizaremos el proceso de limpieza teniendo en cuenta las situaciones más comunes:

1. Datos faltantes en algunas celdas
2. Columnas irrelevantes (que no responden al problema que queremos resolver)
3. Registros (filas) repetidos
4. Valores extremos (outliers) en el caso de las variables numéricas. Se deben analizar en detalle pues no necesariamente la solución es eliminarlos.
5. Errores tipográficos en el caso de las variables categóricas.

Se debe planear de manera adecuada cuales son los pasos a seguir para realizar la limpieza de los datos.

Negocio: *Prueba de Conocimiento Datos no Estructurados*

- Desarrollo – Limpieza de datos – Columnas irrelevantes:

Identifique que la mayoría de columnas del Dataframe son Categóricas, así que las filtré para poder identificar que tantos datos tenían y cuales pueden servir.

-Columnas Irrelevantes

```
#Se genera una lista con todas los nombres de las columnas
col_precios = list(precios.columns.values)

#Se genera lista con todas las columnas con datos categóricas
col_categ_precios = []
for col in col_precios:
    if precios[col].dtype == "object":
        col_categ_precios.append(col)

#Conteo de los niveles en las diferentes columnas categóricas
for colg in col_categ_precios:
  print(f'Columna {colg}: {precios[colg].nunique()} subniveles')
```

```
Columna objeto: 4 subniveles
Columna motivo: 22 subniveles
Columna proposito: 8 subniveles
Columna tipo_avaluo: 3 subniveles
Columna tipo_credito: 4 subniveles
Columna tipo_subsidio: 5 subniveles
Columna departamento_inmueble: 53 subniveles
Columna municipio_inmueble: 444 subniveles
Columna barrio: 6103 subniveles
Columna sector: 12 subniveles
Columna direccion_inmueble_informe: 12826 subniveles
Columna alcantarillado_en_el_sector: 6 subniveles
Columna acueducto_en_el_sector: 4 subniveles
Columna gas_en_el_sector: 3 subniveles
Columna energia_en_el_sector: 2 subniveles
Columna telefono_en_el_sector: 2 subniveles
```

Negocio: *Prueba de Conocimiento Datos no Estructurados*

- Desarrollo – Limpieza de datos – Columnas irrelevantes:

```
#Se identifica visualmente que las columnas importantes tienen más de 226
columnas_precios = []

for colg in col_categ_precios:
    if precios[colg].nunique() < 227:
        columnas_precios.append(colg)

print(columnas_precios)
```

No identifique alguna forma de saber que datos son importantes debido a la gran cantidad de los mismos, así que realicé una lectura de los datos rápidamente para identificar que la mayoría de datos con los que podía trabajar tenían más de 226 subniveles.

```
#Se elimiinan las columnas innecesarias
for i in columnas_precios:
    precios = precios.drop(columns=[i])
```

Luego de identificación, procedí a eliminar los datos que no necesitaba

Negocio: *Prueba de Conocimiento Datos no Estructurados*

- Desarrollo – Limpieza de datos – Columnas irrelevantes:

```
#Se revisa las columnas números
precios.describe()
```

	Unnamed: 0	id	fecha_aprobación	bano_social	bano_servicio	cocina	estudio	balcon	terraza	patio_interior	...	oficina	1285
count	12857.00000	12857.000000	5.344000e+03	12857.000000	12857.000000	12857.000000	12857.000000	12857.000000	12857.000000	12857.000000	...	12857.000000	
mean	6428.00000	9182.685852	4.335527e+04	1.021778	0.136579	1.065879	0.245469	0.510695	0.211947	0.382204	...	0.010967	
std	3711.64054	5290.674433	2.179096e+04	0.753539	0.377717	0.423449	0.440563	0.663919	0.465473	0.587005	...	0.201784	
min	0.00000	1.000000	4.297933e+04	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	
25%	3214.00000	4621.000000	4.301044e+04	1.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	
50%	6428.00000	9182.000000	4.304671e+04	1.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	
75%	9642.00000	13770.000000	4.308845e+04	1.000000	0.000000	1.000000	0.000000	1.000000	0.000000	1.000000	...	0.000000	
max	12856.00000	18331.000000	1.636029e+06	14.000000	11.000000	13.000000	3.000000	11.000000	9.000000	11.000000	...	9.000000	

8 rows × 24 columns

Procedí a verificar las columnas con datos números.

Negocio: *Prueba de Conocimiento Datos no Estructurados*

- Desarrollo – Limpieza de datos – Columnas irrelevantes:

```
#Se revisa la columna Unnamed y id
print(precios["Unnamed: 0"].nunique()) #Se interpreta que la columna es el # de fila
print(precios["id"].nunique())
```

1)

```
12857
12857
```

```
#Se terminarn de eliminan las columnas innecesarias
precios = precios.drop(columns=["Latitud"])
precios = precios.drop(columns=["Longitud"])
precios = precios.drop(columns=["matricula_inmobiliaria_deposito_5"])
precios = precios.drop(columns=["numero_deposito_5"])
precios = precios.drop(columns=["matricula_inmobiliaria_deposito_4"])
precios = precios.drop(columns=["matricula_inmobiliaria_deposito_3"])
precios = precios.drop(columns=["matricula_garaje_5"])
precios = precios.drop(columns=["numero_garaje_5"])
precios = precios.drop(columns=["fecha_aprobación"])
```

2)

1) Identifiqué que habían dos columnas con mucha cantidad de datos así que pase a revisarlas.

2) Procedía re visar la información e identifique columnas que no tenían datos relevantes para la ejecución de la prueba y procedí a eliminarlos.

Negocio: *Prueba de Conocimiento Datos no Estructurados*

- Desarrollo – Limpieza de datos – Columnas irrelevantes:

In [16]:
```
print(precios["descripcion_clase_inmueble"].unique())
```

```
['0' '2 plantas arquitectonicas' 'Ubicado en conjunto cerrado' ...
 'EN  CONJUNTO' 'Multifamiliar, sin ascensor.'
 'alcoba ppal con baÃto privado']
```

In [17]:
```
#Se revisa con el código anterior las columnas restante para veriificar si son utiles
precios = precios.drop(columns=["concepto_del_metodo_5"])
precios = precios.drop(columns=["concepto_del_metodo_8"])
precios = precios.drop(columns=["matricula_inmobiliaria_deposito_1"])
precios = precios.drop(columns=["numero_deposito_1"])
precios = precios.drop(columns=["matricula_garaje_2"])
precios = precios.drop(columns=["numero_garaje_2"])
precios = precios.drop(columns=["matricula_garaje_1"])
precios = precios.drop(columns=["numero_garaje_1"])
precios = precios.drop(columns=["observaciones_generales_inmueble"])
precios = precios.drop(columns=["area_actividad"])
precios = precios.drop(columns=["area_valorada"])
precios = precios.drop(columns=["observaciones_indice_construccion"])
```

Revisé las últimas columnas en las que tenía dudas de manera manual con la función de .unique() y luego procedí a eliminarlas.

Negocio: *Prueba de Conocimiento Datos no Estructurados*

- Desarrollo – Limpieza de datos – Columnas irrelevantes:

```
[19]: #Organizar datos
      precios.sort_values(by="Unnamed: 0", inplace=True)
      precios.head(10)
```

[19]:

	Unnamed: 0	id	municipio_inmueble	barrio	direccion_inmueble_informe	descripcion_general_sector	perspectivas_de_valorizacion	actualidad_edificadora	comportami
819	0	1	SOACHA	BUENOS AIRES	"KR 7 C # 2 A - 30 SUR CS 2"	"Vivienda multifamiliar-zonas verdes-arborizac...	"Teniendo en cuenta el estado del inmueble y s...	"Es un sector consolidado, al momento de la vi...	
9661	1	3	SINCELEJO	LAS FLORES	"KR 6 # 6 - 20"	"El sector donde se localiza el inmueble es de...	"Normales, sector consolidado."	"En la actualidad, se observan que no se están...	"Tanto la o
7291	2	4	CALI	FLORA INDUSTRIAL	"KR 7 # 58 - 05 AP 201"	"Sector consolidado de la ciudad."	"Positiva, dada la consolidación de uso reside...	"En el sector no se observan proyectos en etap...	"Presenta
1129	3	5	PASTO	LAS BRISAS	"KR 18 Este # 21 G Bis - 03 MZ 16 CS 1"	"El Barrio Las Brisas es un sector destinado a...	"El mercado inmobiliario del entorno urbano y ...	"En la actualidad, no se observan en el sector...	"La

Procedí a organizar el DataFrame de manera ascendente para terminar la fase de limpieza de datos.

Negocio: *Prueba de Conocimiento Datos no Estructurados*

- Desarrollo – Análisis:

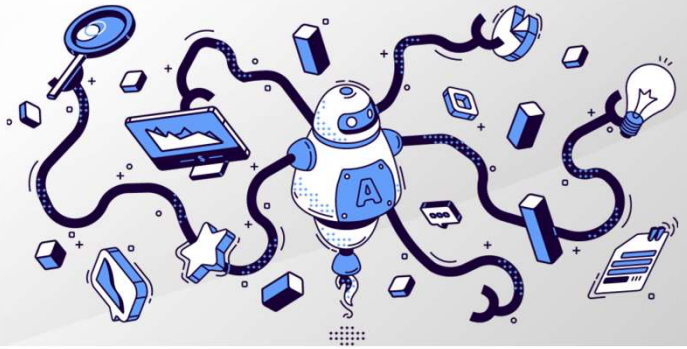
Identificar la cantidad de "habitaciones"

```
col_precios = list(precios.columns.values)

#Se genera lista con todas las columnas con datos categóricas
col_categ = []
for col in col_precios:
    if precios[col].dtype == "int64":
        col_categ.append(col)

col_categ.remove("Unnamed: 0")
col_categ.remove("id")

print(col_categ)
```

```
['bano_social', 'bano_servicio', 'cocina', 'estudio', 'balcon', 'terraza', 'patio_interior', 'jardin', 'zona_de_ropas', 'zona_verde_privada', 'loca
l', 'oficina', 'bodega']
```

```
# Gráficos de barras de conteo
fig, ax = plt.subplots(nrows=len(col_categ), ncols=1, figsize=(10,45))
fig.subplots_adjust(hspace=0.5)

for i, col in enumerate(col_categ):
  sns.countplot(x=col, data=precios, ax=ax[i])
  ax[i].set_title(col)
  ax[i].set_xticklabels(ax[i].get_xticklabels(),rotation=30)
```

Realicé un código para identificar cual era la concentración de habitación de los datos.



Negocio: *Prueba de Conocimiento Datos no Estructurados*

- Desarrollo – Análisis:

Identificar la cantidad de "habitaciones"

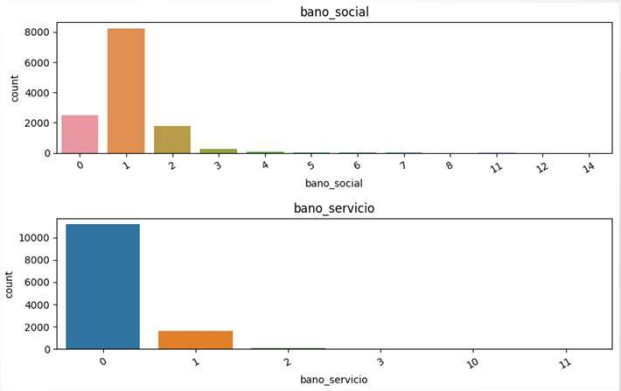
```
col_precios = list(precios.columns.values)

#Se genera lista con todas las columnas con datos categóricas
col_categ = []
for col in col_precios:
    if precios[col].dtype == "int64":
        col_categ.append(col)

col_categ.remove("Unnamed: 0")
col_categ.remove("id")

print(col_categ)
```

```
['bano_social', 'bano_servicio', 'cocina', 'estudio', 'balcon', 'terraza', 'patio_interior', 'jardin', 'zona_de_ropas', 'zona_verde_privada', 'local', 'oficina', 'bodega']
```

```
# Gráficos de barras de conteo
fig, ax = plt.subplots(nrows=len(col_categ), ncols=1, figsize=(10,45))
fig.subplots_adjust(hspace=0.5)

for i, col in enumerate(col_categ):
  sns.countplot(x=col, data=precios, ax=ax[i])
  ax[i].set_title(col)
  ax[i].set_xticklabels(ax[i].get_xticklabels(),rotation=30)
```

Se identifica que en la mayoría de habitaciones que tiene las propiedades, estas no superan 3 habitaciones repetidas.

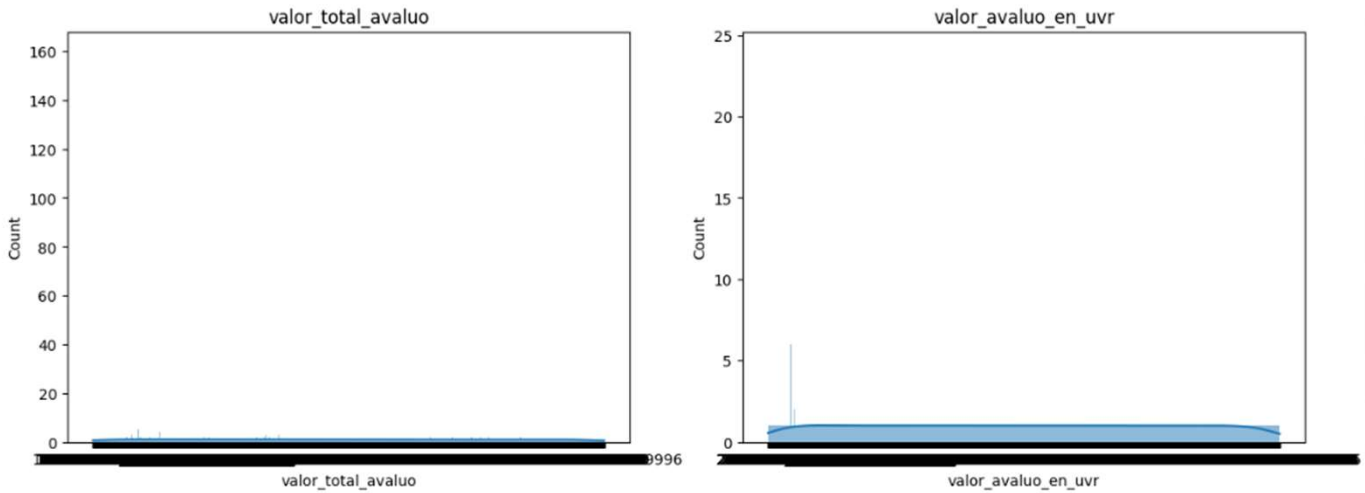
Realicé un código para identificar cual era la concentración de habitación de los datos.



Negocio: *Prueba de Conocimiento Datos no Estructurados*

- Desarrollo – Analisis:

```
#Verificar distribución de los precios
col_num = ['valor_total_avaluo', 'valor_avaluo_en_uvr']

fig, ax = plt.subplots(nrows=1, ncols=2, figsize=(15,5))
fig.subplots_adjust(hspace=0.5)

for i, col in enumerate(col_num):
    if col == 'valor_total_avaluo':
        nbins = 10
    else:
        nbins = 50
    sns.histplot(x=col, data=precios, ax=ax[i], bins=nbins, kde = True)
    ax[i].set_title(col)
```

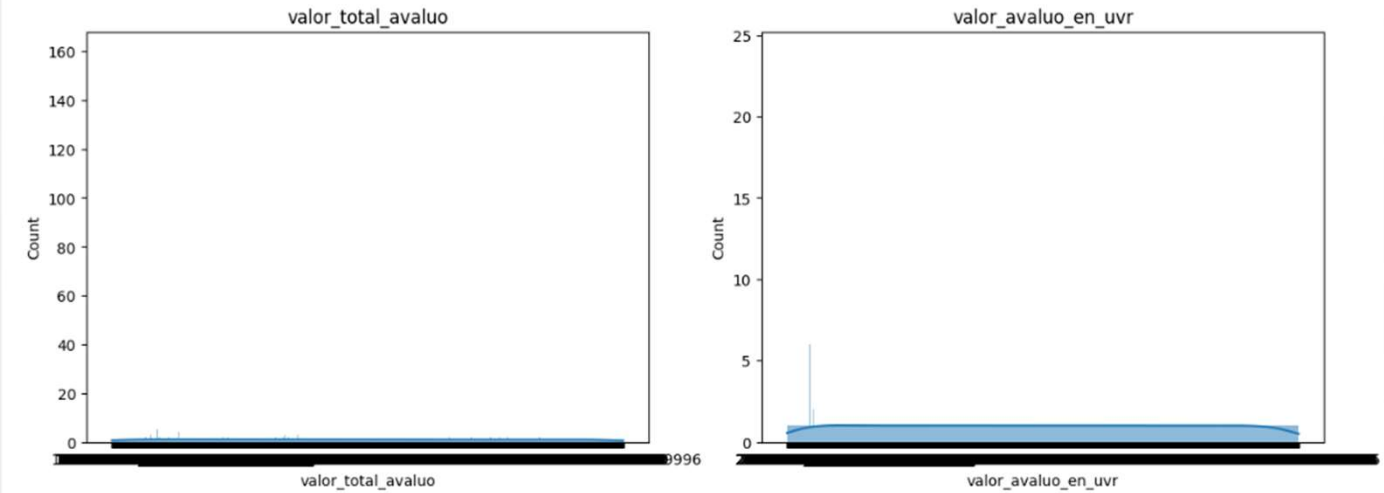


Se realiza un grafico para identificar el comportamiento de los precios

Negocio: *Prueba de Conocimiento Datos no Estructurados*

- Desarrollo – Análisis:

```
#Verificar distribución de los precios
col_num = ['valor_total_avaluo', 'valor_avaluo_en_uvr']

fig, ax = plt.subplots(nrows=1, ncols=2, figsize=(15,5))
fig.subplots_adjust(hspace=0.5)

for i, col in enumerate(col_num):
    if col == 'valor_total_avaluo':
        nbins = 10
    else:
        nbins = 50
    sns.histplot(x=col, data=precios, ax=ax[i], bins=nbins, kde = True)
    ax[i].set_title(col)
```



Se realiza un grafico para identificar el comportamiento de los precios.

Negocio: *Prueba de Conocimiento Datos no Estructurados*

- Desarrollo – Recomendaciones:

Pasos que me faltaron:

-Identificar cuales eran lo predios con mejor rentabilidad.
-Identificar rasgos similares, o algún patrón para poder empezar a realizar un análisis más detallado

Recomendaciones:

-Las decisiones que se deberían tomar luego de predecir el precio de los avalúos es centrarse en aquello que tengan las mejores prestaciones y condiciones para el negocio.

Dificultades:

-Tuve dificultades en el tiempo de realización de la prueba debido a que estuve full en el trabajo y también tenía que asistir a clase, solo tuve algo de tiempo el jueves en horas de la noche.
-Tuve dificultades algunas veces en el manejo de la información ya que no sabía que formula o solución era la más adecuada en esos momentos.


¡Gracias por su atención!