EXPLORACIÓN PROPERATTI DATASET

**Cantidad de filas:** 121.220   
**Cantidad de columnas:** 26  
*Code: df.shape*

**Columnas**:  
*Code: df.columns*['Unnamed: 0', 'operation', 'property\_type', 'place\_name', 'place\_with\_parent\_names', 'country\_name', 'state\_name', 'geonames\_id', 'lat-lon', 'lat', 'lon', 'price', 'currency', 'price\_aprox\_local\_currency', 'price\_aprox\_usd', 'surface\_total\_in\_m2', 'surface\_covered\_in\_m2', 'price\_usd\_per\_m2', 'price\_per\_m2', 'floor', 'rooms', 'expenses', 'properati\_url', 'description', 'title', 'image\_thumbnail']

**Tipos de propiedades:**   
*Code: df['property\_type'].value\_counts()*apartment: 71.065 (59%)  
house: 40.268 (33%)  
PH: 5.751 (5%)  
store: 4.136 (3%)

**Provincias:***Code: df['state\_name'].value\_counts()*

Capital Federal 32.316 (26,7%)

Bs.As. G.B.A. Zona Norte 25.560 (21,1%)

Bs.As. G.B.A. Zona Sur 13.952 (11,5%)

**95,6% de los registros vienen de estas zonas.**

Córdoba 12.069 (10%)

Santa Fe 10.172 (8,4%)

Buenos Aires Costa Atlántica 10.006 (8,3%)

Bs.As. G.B.A. Zona Oeste 9.322 (7,7%)

Buenos Aires Interior 2.291 (1,9%)

Río Negro 808

Neuquén 733

Mendoza 681

Tucumán 674

Corrientes 583

Misiones 464

Entre Ríos 369

Salta 278

Chubut 259

San Luis 252

La Pampa 157

Formosa 65

Chaco 57

San Juan 40

Tierra Del Fuego 31

Catamarca 27

Jujuy 26

Santa Cruz 20

La Rioja 4

Santiago Del Estero 4

**El promedio del precio por m2 varía mucho según el tipo de propiedad y la provincia.   
Dentro de Capital Federal el precio varía mucho por barrio.**

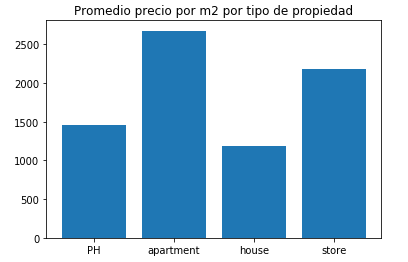
*Codes:*

*promedio\_precio\_m2\_property\_type = df['price\_usd\_per\_m2'].groupby(df['property\_type']).describe()*

*promedio\_precio\_m2\_property\_type.reset\_index(inplace=True)*

*plt.bar(promedio\_precio\_m2\_property\_type['property\_type'], promedio\_precio\_m2\_property\_type['mean'])*

*plt.title('Promedio precio por m2 por tipo de propiedad'))*



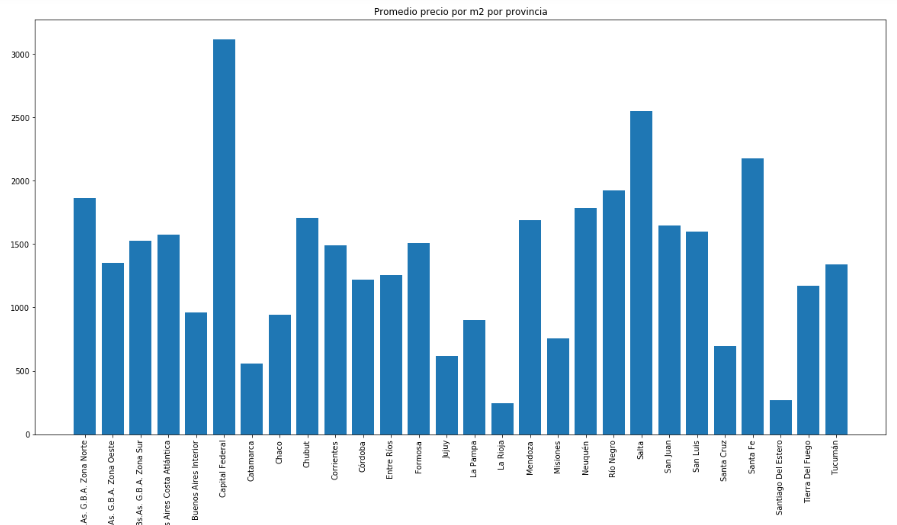
*promedio\_precio\_m2\_provincia = df['price\_usd\_per\_m2'].groupby(df['state\_name']).describe()*

*promedio\_precio\_m2\_provincia.reset\_index(inplace=True)*

*plt.figure(figsize=(20,10))*

*plt.bar(promedio\_precio\_m2\_provincia['state\_name'], promedio\_precio\_m2\_provincia['mean'])*

*plt.title('Promedio precio por m2 por provincia')*

*plt.xticks(rotation='vertical')*

*promedio\_precio\_m2\_barrio = df['price\_usd\_per\_m2'][(df['state\_name'] == 'Capital Federal')].groupby(df['place\_name']).describe()*

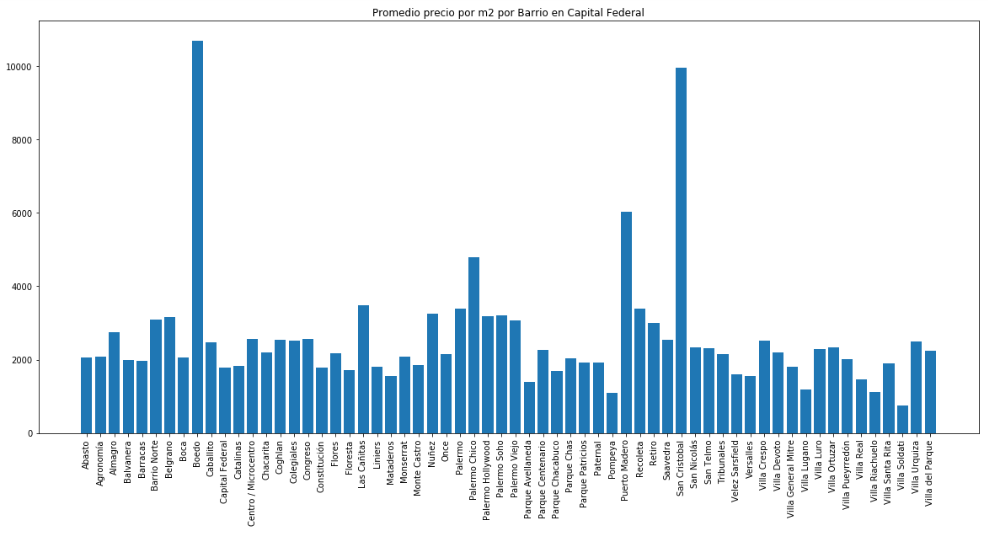
*promedio\_precio\_m2\_barrio.reset\_index(inplace=True)*

*plt.figure(figsize=(20,10))*

*plt.bar(promedio\_precio\_m2\_barrio['place\_name'], promedio\_precio\_m2\_barrio['mean'])*

*plt.title('Promedio precio por m2 por Barrio en Capital Federal')*

*plt.xticks(rotation='vertical')*

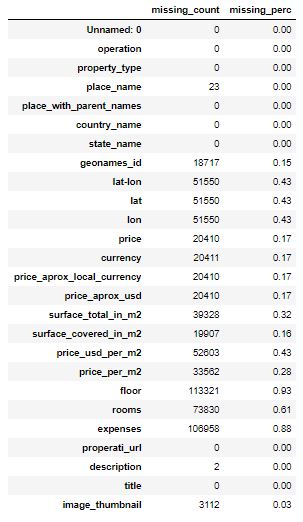


PARA LIMPIAR

**Hay 1.222 filas duplicadas** (1% del dataset) 🡪 **Borrar?** O puede ser que por ejemplo hay varios departamentos en el mismo edificio?  
*Code: df.loc[:, ~df.columns.isin(['Unnamed: 0','properati\_url'])].duplicated().sum()   
Para ver las filas duplicadas: df[(df.loc[:, ~df.columns.isin(['Unnamed: 0','properati\_url'])].duplicated() == True)]*

**Missing values 🡪** Creo que deberíamos elegir una estrategia por cada columna con datos faltantes. Queremos reemplazar el valor con alguna estadística? (en el caso de los precios y superficies faltantes tal vez podemos reemplazar por precio y superficie promedio de un groupby por zona/barrio y property type?). O podríamos sacar información de otras columnas para completar los datos faltantes? (Descripción/Titulo?) O queremos borrar algunas filas enteras?

*Code:   
missing = df.isnull().sum()  
missing\_perc = round((missing / len(df)),2)  
missing\_df = pd.concat([missing, missing\_perc], axis=1)  
missing\_df.columns = ['missing\_count', 'missing\_perc']*



**Outliers 🡪 Tenemos que elegir que hacer con los outliers. Verificar si son datos reales o si son valores puestos por error. Borrar? Reemplazar por un valor máximo/mínimo?**

*Code:  
def cantidad\_outliers(column):  
 media = df[column].mean()  
 desvio = df[column].std()  
 return len(df.loc[df[column]>(media +3\*desvio)])*

*columns = df.columns.to\_series()*

*for i in columns[['price','price\_aprox\_local\_currency','price\_aprox\_usd','surface\_total\_in\_m2','surface\_covered\_in\_m2','price\_usd\_per\_m2','price\_per\_m2','floor','rooms','expenses']]:  
 print(str(i)+':', cantidad\_outliers(i))*

Cantidad de outliers por variable cuantitativa:

price: 138  
price\_aprox\_local\_currency: 1225  
price\_aprox\_usd: 1225  
surface\_total\_in\_m2: 167  
surface\_covered\_in\_m2: 163  
price\_usd\_per\_m2: 863  
price\_per\_m2: 287  
floor: 90  
rooms: 712  
expenses: 10