2) Machine learning

August 8, 2023

1 1) Preparación previa

1.0.1 Carga de librerías

[1]:
```
import pandas as pd
import numpy as np
import re
from sklearn import linear_model
from sklearn.metrics import mean_squared_error, r2_score, mean_absolute_error
from sklearn.model_selection import train_test_split
%matplotlib inline
from matplotlib import pyplot as plt
import seaborn as sns
import scipy as sp
from sklearn.preprocessing import PolynomialFeatures
from sklearn.model_selection import train_test_split
```

1.0.2 Lectura del data set limpio

[2]:
```
data = pd.read_csv("data_final.csv", sep = ";")
data.head(5)
```

[2]:
```
  property_type     price  surface_covered_in_m2        Partido  \
0            PH   62000.0                   40.0      Mataderos
1     apartment   72000.0                   55.0      Mataderos
2     apartment   64000.0                   35.0  Mar del Plata
3            PH  130000.0                   78.0  Vicente López
4     apartment  138000.0                   40.0       Belgrano

   precio_usd_por_m2  ambientes_train  ambientes_imputados  ambientes_final  \
0        1550.000000                2                  0.0              2.0
1        1309.090909                2                  0.0              2.0
2        1828.571429                2                  0.0              2.0
3        1666.666667                0                  3.0              3.0
4        3450.000000                0                  1.0              1.0

  balcon parrilla  pileta  patio quincho gimnasio sala_usos_multiples cochera  \
```

```
0    NaN      NaN     NaN  patio     NaN      NaN                 NaN     NaN
1    NaN      NaN     NaN    NaN     NaN      NaN                 NaN     NaN
2    NaN      NaN     NaN    NaN     NaN      NaN                 NaN     NaN
3    NaN      NaN     NaN  patio     NaN      NaN                 NaN     NaN
4    NaN      NaN  pileta    NaN     NaN      NaN                 NaN     NaN

  seguridad jardin  frente
0       NaN    NaN     NaN
1       NaN    NaN     NaN
2       NaN    NaN  frente
3       NaN    NaN  frente
4       NaN    NaN     NaN
```

2 2) Creación de dummies

2.0.1 Partido

[3]:
```
# Como hay muchísimos, se usará el dataset con solo los 50 partidos con más␣
↪datos
grupo_partidos = data.groupby("Partido")
filtro_partidos = grupo_partidos[["Partido"]].describe()
data_partidos = filtro_partidos[filtro_partidos.Partido.freq > 289]
len(data_partidos)
```

[3]:
```
50
```

[4]:
```
mask_partidos = data['Partido'].isin(data_partidos.Partido.top)
data_para_dummies = data[mask_partidos]
data_para_dummies
```

[4]:
```
      property_type     price  surface_covered_in_m2        Partido  \
2         apartment   64000.0                   35.0  Mar del Plata
3                PH  130000.0                   78.0  Vicente López
4         apartment  138000.0                   40.0       Belgrano
5         apartment  195000.0                   60.0       Belgrano
6         apartment  115000.0                   36.0     San Isidro
…               …       …                      …             …
71761         house  170000.0                  130.0          Pilar
71762     apartment  128000.0                   35.0       Belgrano
71763     apartment  165000.0                   39.0       Recoleta
71764     apartment  131500.0                   39.0  Villa Urquiza
71765     apartment   95900.0                   48.0  Mar del Plata

       precio_usd_por_m2  ambientes_train  ambientes_imputados  \
2            1828.571429                2                  0.0
3            1666.666667                0                  3.0
4            3450.000000                0                  1.0
```

```
5            3250.000000                0                   2.0
6            3194.444444                2                   0.0
…                  …                  …                     …
71761        1307.692308                0                   4.0
71762        3657.142857                0                   1.0
71763        4230.769231                0                   1.0
71764        3371.794872                0                   1.0
71765        1997.916667                2                   0.0

       ambientes_final  balcon  parrilla  pileta  patio quincho  gimnasio  \
2                  2.0     NaN       NaN     NaN    NaN     NaN       NaN
3                  3.0     NaN       NaN     NaN  patio     NaN       NaN
4                  1.0     NaN       NaN  pileta    NaN     NaN       NaN
5                  2.0     NaN       NaN  pileta    NaN     NaN       NaN
6                  2.0  balcon       NaN     NaN    NaN     NaN       NaN
…                  …       …         …       …      …       …         …
71761              4.0     NaN  parrilla  pileta    NaN     NaN       NaN
71762              1.0     NaN  parrilla     NaN    NaN     NaN  gimnasio
71763              1.0     NaN  parrilla     NaN    NaN     NaN       NaN
71764              1.0  balcon  parrilla     NaN    NaN     NaN       NaN
71765              2.0     NaN       NaN     NaN    NaN     NaN       NaN

      sala_usos_multiples  cochera seguridad jardin  frente
2                     NaN      NaN       NaN    NaN  frente
3                     NaN      NaN       NaN    NaN  frente
4                     NaN      NaN       NaN    NaN     NaN
5                     NaN      NaN       NaN    NaN     NaN
6                     NaN      NaN       NaN    NaN  frente
…                     …        …         …      …       …
71761                 NaN      NaN       NaN    NaN     NaN
71762                 sum  cochera       NaN    NaN  frente
71763                 NaN      NaN       NaN    NaN  frente
71764                 NaN  cochera       NaN    NaN     NaN
71765                 NaN      NaN       NaN    NaN  frente

[62621 rows x 19 columns]
```

[5]:
```
dummy_partido = pd.get_dummies(data_para_dummies['Partido'], prefix='partido')
dummy_partido
```

[5]:
```
       partido_Almagro  partido_Almirante Brown  partido_Avellaneda  \
2                    0                        0                   0
3                    0                        0                   0
4                    0                        0                   0
5                    0                        0                   0
6                    0                        0                   0
…                  …                        …                   …
```

```
71761                0                       0                    0
71762                0                       0                    0
71763                0                       0                    0
71764                0                       0                    0
71765                0                       0                    0

       partido_Bahía Blanca  partido_Balvanera  partido_Barracas  \
2                         0                  0                 0
3                         0                  0                 0
4                         0                  0                 0
5                         0                  0                 0
6                         0                  0                 0
…                       …                  …                 …
71761                     0                  0                 0
71762                     0                  0                 0
71763                     0                  0                 0
71764                     0                  0                 0
71765                     0                  0                 0

       partido_Barrio Norte  partido_Belgrano  partido_Boedo  \
2                         0                 0              0
3                         0                 0              0
4                         0                 1              0
5                         0                 1              0
6                         0                 0              0
…                       …                 …              …
71761                     0                 0              0
71762                     0                 1              0
71763                     0                 0              0
71764                     0                 0              0
71765                     0                 0              0

       partido_Caballito  …  partido_San Telmo  partido_Tigre  \
2                      0  …                  0              0
3                      0  …                  0              0
4                      0  …                  0              0
5                      0  …                  0              0
6                      0  …                  0              0
…                    …  …                  …              …
71761                  0  …                  0              0
71762                  0  …                  0              0
71763                  0  …                  0              0
71764                  0  …                  0              0
71765                  0  …                  0              0

       partido_Tres de Febrero  partido_Vicente López  \
2                            0                      0
```

```
3                              0                       1
4                              0                       0
5                              0                       0
6                              0                       0
…                            …                       …
71761                          0                       0
71762                          0                       0
71763                          0                       0
71764                          0                       0
71765                          0                       0

       partido_Villa Carlos Paz  partido_Villa Crespo  partido_Villa Devoto  \
2                              0                       0                     0
3                              0                       0                     0
4                              0                       0                     0
5                              0                       0                     0
6                              0                       0                     0
…                            …                       …                   …
71761                          0                       0                     0
71762                          0                       0                     0
71763                          0                       0                     0
71764                          0                       0                     0
71765                          0                       0                     0

       partido_Villa Luro  partido_Villa Urquiza  partido_Villa del Parque
2                       0                       0                        0
3                       0                       0                        0
4                       0                       0                        0
5                       0                       0                        0
6                       0                       0                        0
…                     …                       …                      …
71761                   0                       0                        0
71762                   0                       0                        0
71763                   0                       0                        0
71764                   0                       1                        0
71765                   0                       0                        0

[62621 rows x 50 columns]
```

2.0.2 Propiedades

[6]:
```
dummy_prop = pd.get_dummies(data_para_dummies['property_type'], prefix='prop')
dummy_prop
```

[6]:
```
       prop_PH  prop_apartment  prop_house
2            0               1           0
3            1               0           0
```

```
4              0                1              0
5              0                1              0
6              0                1              0
…          …                …              …
71761          0                0              1
71762          0                1              0
71763          0                1              0
71764          0                1              0
71765          0                1              0

[62621 rows x 3 columns]
```

2.0.3 Ambientes

[7]:
```
dummy_amb = pd.get_dummies(data_para_dummies['ambientes_final'], prefix='amb')
dummy_amb
```

[7]:
```
       amb_1.0  amb_2.0  amb_3.0  amb_4.0  amb_5.0  amb_6.0  amb_7.0
2            0        1        0        0        0        0        0
3            0        0        1        0        0        0        0
4            1        0        0        0        0        0        0
5            0        1        0        0        0        0        0
6            0        1        0        0        0        0        0
…        …       …        …        …        …        …        …
71761        0        0        0        1        0        0        0
71762        1        0        0        0        0        0        0
71763        1        0        0        0        0        0        0
71764        1        0        0        0        0        0        0
71765        0        1        0        0        0        0        0

[62621 rows x 7 columns]
```

2.0.4 Balcón

[8]:
```
dummy_balcon = pd.get_dummies(data_para_dummies['balcon'], prefix = "dummy")
dummy_balcon
```

[8]:
```
       dummy_balcon
2                 0
3                 0
4                 0
5                 0
6                 1
…               …
71761             0
71762             0
71763             0
```

```
71764             1
71765             0

[62621 rows x 1 columns]
```

2.0.5 Parrilla

[9]:
```
dummy_parrilla = pd.get_dummies(data_para_dummies['parrilla'], prefix = "dummy")
dummy_parrilla
```

[9]:
```
       dummy_parrilla
2                   0
3                   0
4                   0
5                   0
6                   0
…                   …
71761               1
71762               1
71763               1
71764               1
71765               0

[62621 rows x 1 columns]
```

2.0.6 Pileta

[10]:
```
dummy_pileta = pd.get_dummies(data_para_dummies['pileta'], prefix = "dummy")
dummy_pileta
```

[10]:
```
       dummy_pileta
2                 0
3                 0
4                 1
5                 1
6                 0
…                 …
71761             1
71762             0
71763             0
71764             0
71765             0

[62621 rows x 1 columns]
```

2.0.7 Patio

[11]:
```
dummy_patio = pd.get_dummies(data_para_dummies['patio'], prefix = "dummy")
dummy_patio
```

[11]:
```
       dummy_patio
2                0
3                1
4                0
5                0
6                0
…              …
71761            0
71762            0
71763            0
71764            0
71765            0

[62621 rows x 1 columns]
```

2.0.8 Quincho

[12]:
```
dummy_quincho = pd.get_dummies(data_para_dummies['quincho'], prefix = "dummy")
dummy_quincho
```

[12]:
```
       dummy_quincho
2                  0
3                  0
4                  0
5                  0
6                  0
…                …
71761              0
71762              0
71763              0
71764              0
71765              0

[62621 rows x 1 columns]
```

2.0.9 Gimnasio

[13]:
```
dummy_gimnasio = pd.get_dummies(data_para_dummies['gimnasio'], prefix = "dummy")
dummy_gimnasio
```

[13]:
```
       dummy_gimnasio
2                   0
3                   0
```

```
4                    0
5                    0
6                    0
…                  …
71761                0
71762                1
71763                0
71764                0
71765                0

[62621 rows x 1 columns]
```

2.0.10 SUM

[14]:
```
dummy_sum = pd.get_dummies(data_para_dummies['sala_usos_multiples'], prefix =␣
↪"dummy")
dummy_sum
```

[14]:
```
       dummy_sum
2              0
3              0
4              0
5              0
6              0
…            …
71761          0
71762          1
71763          0
71764          0
71765          0

[62621 rows x 1 columns]
```

2.0.11 Cochera

[15]:
```
dummy_cochera = pd.get_dummies(data_para_dummies['cochera'], prefix = "dummy")
dummy_cochera
```

[15]:
```
       dummy_cochera
2                  0
3                  0
4                  0
5                  0
6                  0
…                  …
71761              0
71762              1
```

```
71763              0
71764              1
71765              0

[62621 rows x 1 columns]
```

2.0.12 Seguridad

[16]:
```
dummy_seguridad = pd.get_dummies(data_para_dummies['seguridad'], prefix =␣
↪"dummy")
dummy_seguridad
```

[16]:
```
       dummy_seguridad
2                    0
3                    0
4                    0
5                    0
6                    0
…                  …
71761                0
71762                0
71763                0
71764                0
71765                0

[62621 rows x 1 columns]
```

2.0.13 Jardín

[17]:
```
dummy_jardin = pd.get_dummies(data_para_dummies['jardin'], prefix = "dummy")
dummy_jardin
```

[17]:
```
       dummy_jardin
2                 0
3                 0
4                 0
5                 0
6                 0
…               …
71761             0
71762             0
71763             0
71764             0
71765             0

[62621 rows x 1 columns]
```

2.0.14 Frente

[18]:
```
dummy_frente = pd.get_dummies(data_para_dummies['frente'], prefix = "dummy")
dummy_frente
```

[18]:
```
       dummy_frente
2                 1
3                 1
4                 0
5                 0
6                 1
…               …
71761             0
71762             1
71763             1
71764             0
71765             1

[62621 rows x 1 columns]
```

[19]:
```
dummy_amenities = dummy_balcon.dummy_balcon + dummy_parrilla.dummy_parrilla +␣
 ↪dummy_pileta.dummy_pileta + dummy_patio.dummy_patio + dummy_quincho.
 ↪dummy_quincho + dummy_gimnasio.dummy_gimnasio + dummy_sum.dummy_sum +␣
 ↪dummy_cochera.dummy_cochera + dummy_seguridad.dummy_seguridad + dummy_jardin.
 ↪dummy_jardin + dummy_frente.dummy_frente
dummy_amenities.name = "dummy_amenities"
```

[20]:
```
dummy_amenities.value_counts()
```

[20]:
```
1     16726
2     15485
0     11093
3      9686
4      5526
5      2665
6       994
7       402
8        38
10        4
9         2
Name: dummy_amenities, dtype: int64
```

2.0.15 Unificación de dummies en un dataset

[21]:
```
data_con_dummies = pd.concat([data_para_dummies, dummy_amenities,␣
 ↪dummy_partido, dummy_prop, dummy_amb],axis=1)
data_con_dummies.columns
```

[21]:
```
Index(['property_type', 'price', 'surface_covered_in_m2', 'Partido',
       'precio_usd_por_m2', 'ambientes_train', 'ambientes_imputados',
       'ambientes_final', 'balcon', 'parrilla', 'pileta', 'patio', 'quincho',
       'gimnasio', 'sala_usos_multiples', 'cochera', 'seguridad', 'jardin',
       'frente', 'dummy_amenities', 'partido_Almagro',
       'partido_Almirante Brown', 'partido_Avellaneda', 'partido_Bahía Blanca',
       'partido_Balvanera', 'partido_Barracas', 'partido_Barrio Norte',
       'partido_Belgrano', 'partido_Boedo', 'partido_Caballito',
       'partido_Colegiales', 'partido_Córdoba', 'partido_Escobar',
       'partido_Esteban Echeverría', 'partido_Ezeiza', 'partido_Flores',
       'partido_Floresta', 'partido_General San Martín', 'partido_Ituzaingó',
       'partido_La Matanza', 'partido_La Plata', 'partido_Lanús',
       'partido_Lomas de Zamora', 'partido_Mar del Plata', 'partido_Monserrat',
       'partido_Moreno', 'partido_Morón', 'partido_Nuñez', 'partido_Palermo',
       'partido_Pilar', 'partido_Pinamar', 'partido_Punilla',
       'partido_Quilmes', 'partido_Recoleta', 'partido_Rosario',
       'partido_Saavedra', 'partido_San Cristobal', 'partido_San Fernando',
       'partido_San Isidro', 'partido_San Miguel', 'partido_San Telmo',
       'partido_Tigre', 'partido_Tres de Febrero', 'partido_Vicente López',
       'partido_Villa Carlos Paz', 'partido_Villa Crespo',
       'partido_Villa Devoto', 'partido_Villa Luro', 'partido_Villa Urquiza',
       'partido_Villa del Parque', 'prop_PH', 'prop_apartment', 'prop_house',
       'amb_1.0', 'amb_2.0', 'amb_3.0', 'amb_4.0', 'amb_5.0', 'amb_6.0',
       'amb_7.0'],
      dtype='object')
```

3 3) Creación de modelo uninominal

3.1 Aplicación en todo el dataset

[22]:
```
X_train_todo_dataset = data_con_dummies[data_con_dummies.ambientes_imputados ==␣
↪0].drop(["property_type", "price", "surface_covered_in_m2", "Partido",␣
↪"ambientes_train",
                                                                        ␣
↪"ambientes_imputados", "ambientes_final", "balcon", "parrilla", "pileta",␣
↪"patio",
                                                                        ␣
↪"quincho", "gimnasio", "sala_usos_multiples", "cochera", "seguridad",␣
↪"jardin", "frente"], axis = 1)
X_test_todo_dataset = data_con_dummies[data_con_dummies.ambientes_train == 0].
↪drop(["property_type", "price", "surface_covered_in_m2", "Partido",␣
↪"ambientes_train",
                                                                        ␣
↪"ambientes_imputados", "ambientes_final", "balcon", "parrilla", "pileta",␣
↪"patio",
```

```
 ↪"quincho", "gimnasio", "sala_usos_multiples", "cochera", "seguridad",␣
 ↪"jardin", "frente"], axis = 1)
Y_train_todo_dataset = X_train_todo_dataset["precio_usd_por_m2"]
Y_test_todo_dataset = X_test_todo_dataset["precio_usd_por_m2"]

X_train_todo_dataset.drop(["precio_usd_por_m2"], axis = 1, inplace = True)
X_test_todo_dataset.drop(["precio_usd_por_m2"], axis = 1, inplace = True)
```

[23]:
```
data_con_dummies.shape
```

[23]:
```
(62621, 80)
```

[24]:
```
X_train_todo_dataset.columns
```

[24]:
```
Index(['dummy_amenities', 'partido_Almagro', 'partido_Almirante Brown',
       'partido_Avellaneda', 'partido_Bahía Blanca', 'partido_Balvanera',
       'partido_Barracas', 'partido_Barrio Norte', 'partido_Belgrano',
       'partido_Boedo', 'partido_Caballito', 'partido_Colegiales',
       'partido_Córdoba', 'partido_Escobar', 'partido_Esteban Echeverría',
       'partido_Ezeiza', 'partido_Flores', 'partido_Floresta',
       'partido_General San Martín', 'partido_Ituzaingó', 'partido_La Matanza',
       'partido_La Plata', 'partido_Lanús', 'partido_Lomas de Zamora',
       'partido_Mar del Plata', 'partido_Monserrat', 'partido_Moreno',
       'partido_Morón', 'partido_Nuñez', 'partido_Palermo', 'partido_Pilar',
       'partido_Pinamar', 'partido_Punilla', 'partido_Quilmes',
       'partido_Recoleta', 'partido_Rosario', 'partido_Saavedra',
       'partido_San Cristobal', 'partido_San Fernando', 'partido_San Isidro',
       'partido_San Miguel', 'partido_San Telmo', 'partido_Tigre',
       'partido_Tres de Febrero', 'partido_Vicente López',
       'partido_Villa Carlos Paz', 'partido_Villa Crespo',
       'partido_Villa Devoto', 'partido_Villa Luro', 'partido_Villa Urquiza',
       'partido_Villa del Parque', 'prop_PH', 'prop_apartment', 'prop_house',
       'amb_1.0', 'amb_2.0', 'amb_3.0', 'amb_4.0', 'amb_5.0', 'amb_6.0',
       'amb_7.0'],
      dtype='object')
```

Creación de los elementos a utilizar

[25]:
```
lm_todo_dataset = linear_model.LinearRegression()
# Fiteamos el modelo sobre los vectores X e Y.
model_todo_dataset = lm_todo_dataset.fit(X_train_todo_dataset,␣
 ↪Y_train_todo_dataset)
```

Aplicación del modelo en los datos

[26]:
```
# Guardamos  las predicciones en un nuevo vector que llamaremos predictions.
predictions_todo_dataset = lm_todo_dataset.predict(X_test_todo_dataset)
```

```
# Imprimimos el intercepto y los coeficientes como atributos del objeto␣
 ↪entrenado.
print ('Intercepto =', model_todo_dataset.intercept_)
print ('RM =', model_todo_dataset.coef_)

# Imprimos la metrica que mide la bondad de ajusto del modelo. En este caso el␣
 ↪R2.
print ('R2_train =', model_todo_dataset.score(X_train_todo_dataset,␣
 ↪Y_train_todo_dataset))
```

```
Intercepto = -275374778319223.25
RM = [ 6.48698166e+01  7.51959270e+13  7.51959270e+13  7.51959270e+13
  7.51959270e+13  7.51959270e+13  7.51959270e+13  7.51959270e+13
  7.51959270e+13  7.51959270e+13  7.51959270e+13  7.51959270e+13
  7.51959270e+13  7.51959270e+13  7.51959270e+13  7.51959270e+13
  7.51959270e+13  7.51959270e+13  7.51959270e+13  7.51959270e+13
  7.51959270e+13  7.51959270e+13  7.51959270e+13  7.51959270e+13
  7.51959270e+13  7.51959270e+13  7.51959270e+13  7.51959270e+13
  7.51959270e+13  7.51959270e+13  7.51959270e+13  7.51959270e+13
  7.51959270e+13  7.51959270e+13  7.51959270e+13  7.51959270e+13
  7.51959270e+13  7.51959270e+13  7.51959270e+13  7.51959270e+13
  7.51959270e+13  7.51959270e+13  7.51959270e+13  7.51959270e+13
  7.51959270e+13  7.51959270e+13  7.51959270e+13  7.51959270e+13
  7.51959270e+13  7.51959270e+13  7.51959270e+13 -1.69280267e+13
 -1.69280267e+13 -1.69280267e+13  2.17106878e+14  2.17106878e+14
  2.17106878e+14  2.17106878e+14  2.17106878e+14  2.17106878e+14
  2.17106878e+14]
R2_train = 0.5425370943748955
```

Obtención de métricas

[27]:
```
#media del precio_usd_por_m2
data_con_dummies.precio_usd_por_m2.mean()
```

[27]: 2144.7300796479353

[28]:
```
# Error absoluto de la media
mean_absolute_error(Y_test_todo_dataset, predictions_todo_dataset)
```

[28]: 397.77032915468135

[29]:
```
# Raiz cuadrada de la media del error
mean_squared_error(Y_test_todo_dataset, predictions_todo_dataset)
```

[29]: 295348.7421802869

[30]:
```
# Raiz cuadrada de la media del error
RMSE = np.sqrt(mean_squared_error(Y_test_todo_dataset,␣
 ↪predictions_todo_dataset))
```

```
RMSE
```

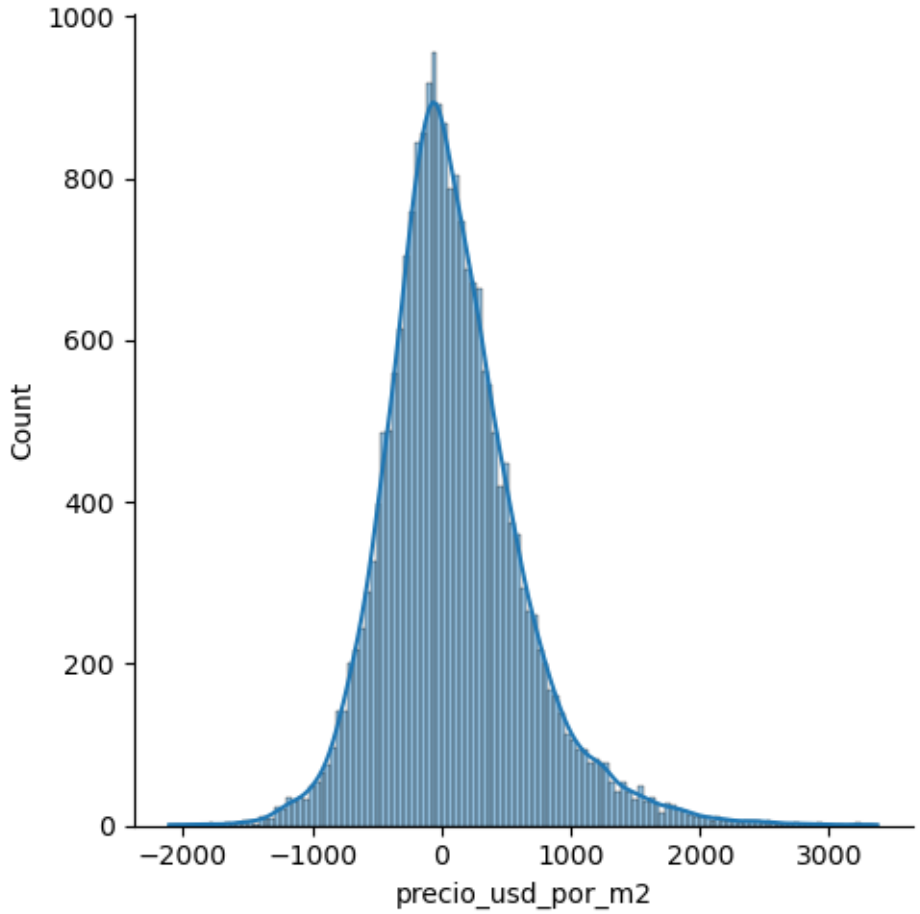
[30]:
```
543.4599729329539
```

Análisis para Regresion Lineal En primer lugar, se obtiene el conjunto de datos residuales y se los grafica para observar su distribución.

[31]:
```
test_residuals = Y_test_todo_dataset - predictions_todo_dataset
```

[32]:
```
sns.displot(test_residuals, kde = True)
```

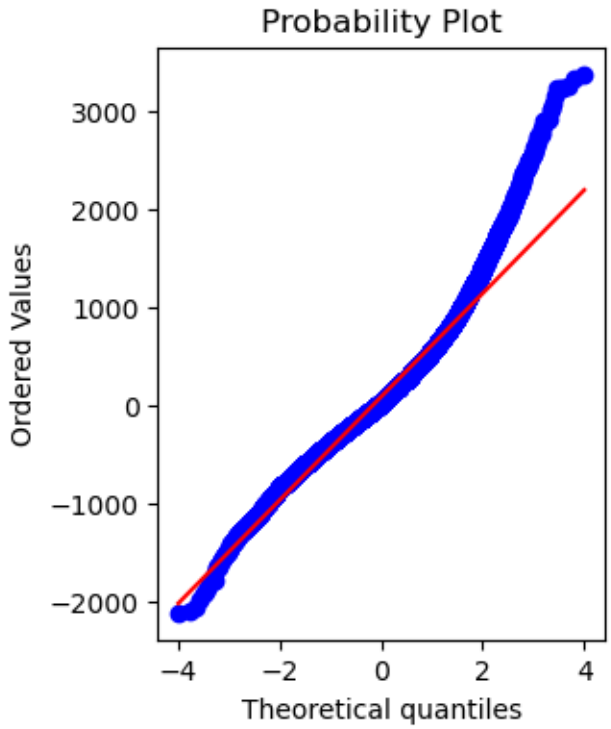
[32]:
```
<seaborn.axisgrid.FacetGrid at 0x27e00031a90>
```



La distribucion de los Errores Residuales pareciera ser una Normal con media en 0. Igualmente, podría haber ser un caso del Cuarteto de Anscombe por lo que se procede graficando la probabilidad en torno a los quantiles.

[33]:
```
fig,ax = plt.subplots(figsize = (3,4), dpi = 100)
sp.stats.probplot(test_residuals,plot = ax)
```

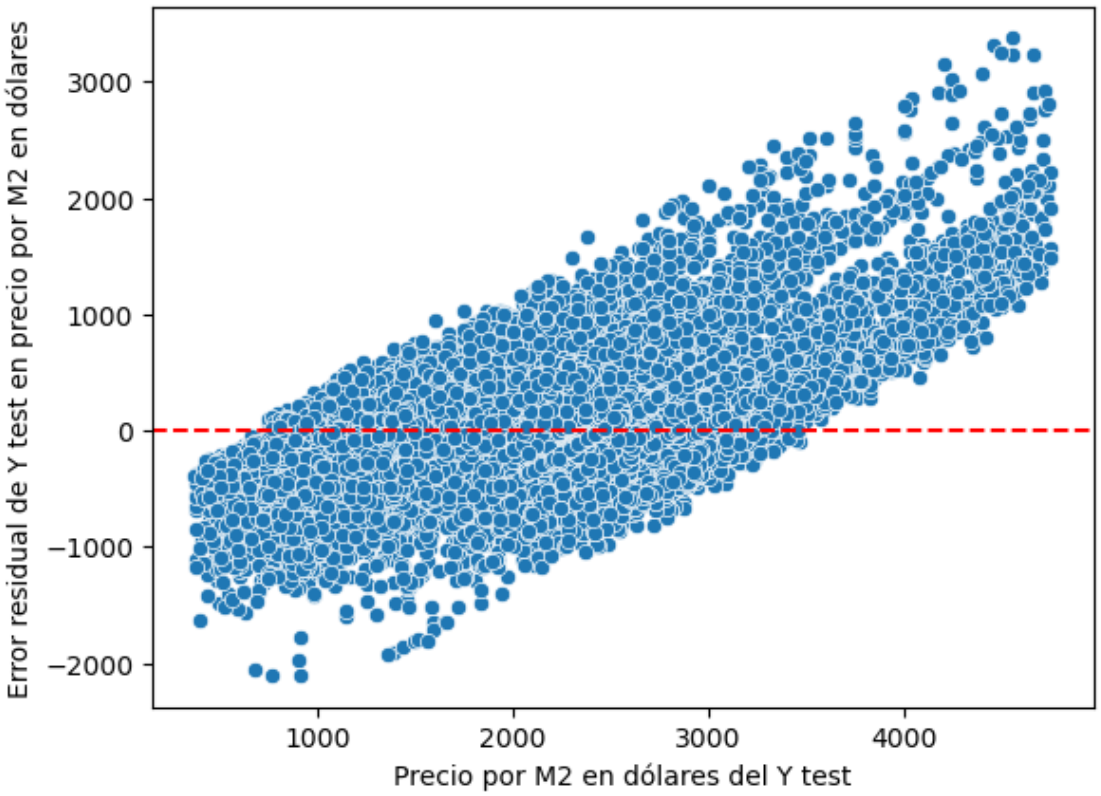
[33]:
```
((array([-4.00199854, -3.7870279 , -3.66944762, …,  3.66944762,
          3.7870279 ,  4.00199854]),
  array([-2108.47159091, -2098.44387755, -2051.33894231, …,
          3256.13574735,  3323.62946429,  3374.11875   ])),
 (525.8168880729593, 89.6324458566893, 0.9807982984381011))
```



Con este gráfico se corrobora efectivamente que la distribución de los Errores Residuales se aproxima a una Distribución Normal y, por ende, se puede proceder con la creación de la regresion lineal.

[34]:
```
sns.scatterplot(x = Y_test_todo_dataset, y = test_residuals)
plt.axhline(y = 0, color = 'r', ls = "--")
plt.xlabel("Precio por M2 en dólares del Y test")
plt.ylabel("Error residual de Y test en precio por M2 en dólares")
```

[34]:
```
Text(0, 0.5, 'Error residual de Y test en precio por M2 en dólares')
```



3.2 Regresión lineal entre superficie cubierta y precio por m2 en dólares

[35]:
```
X_train_superficie = data_con_dummies[data_con_dummies.ambientes_imputados ==␣
↪0].drop(['property_type', 'price', 'Partido', 'ambientes_train',␣
↪'ambientes_imputados',
      'ambientes_final', 'balcon', 'parrilla', 'pileta', 'patio', 'quincho',
      'gimnasio', 'sala_usos_multiples', 'cochera', 'seguridad', 'jardin',
      'frente', 'dummy_amenities', 'partido_Almagro',
      'partido_Almirante Brown', 'partido_Avellaneda', 'partido_Bahía Blanca',
      'partido_Balvanera', 'partido_Barracas', 'partido_Barrio Norte',
      'partido_Belgrano', 'partido_Boedo', 'partido_Caballito',
      'partido_Colegiales', 'partido_Córdoba', 'partido_Escobar',
      'partido_Esteban Echeverría', 'partido_Ezeiza', 'partido_Flores',
      'partido_Floresta', 'partido_General San Martín', 'partido_Ituzaingó',
      'partido_La Matanza', 'partido_La Plata', 'partido_Lanús',
      'partido_Lomas de Zamora', 'partido_Mar del Plata', 'partido_Monserrat',
      'partido_Moreno', 'partido_Morón', 'partido_Nuñez', 'partido_Palermo',
      'partido_Pilar', 'partido_Pinamar', 'partido_Punilla',
      'partido_Quilmes', 'partido_Recoleta', 'partido_Rosario',
      'partido_Saavedra', 'partido_San Cristobal', 'partido_San Fernando',
      'partido_San Isidro', 'partido_San Miguel', 'partido_San Telmo',
```

```
       'partido_Tigre', 'partido_Tres de Febrero', 'partido_Vicente López',
       'partido_Villa Carlos Paz', 'partido_Villa Crespo',
       'partido_Villa Devoto', 'partido_Villa Luro', 'partido_Villa Urquiza',
       'partido_Villa del Parque', 'prop_PH', 'prop_apartment', 'prop_house',
       'amb_1.0', 'amb_2.0', 'amb_3.0', 'amb_4.0', 'amb_5.0', 'amb_6.0',
       'amb_7.0'], axis = 1)
X_test_superficie = data_con_dummies[data_con_dummies.ambientes_train == 0].
 ↪drop(['property_type', 'price', 'Partido', 'ambientes_train',␣
 ↪'ambientes_imputados',
       'ambientes_final', 'balcon', 'parrilla', 'pileta', 'patio', 'quincho',
       'gimnasio', 'sala_usos_multiples', 'cochera', 'seguridad', 'jardin',
       'frente', 'dummy_amenities', 'partido_Almagro',
       'partido_Almirante Brown', 'partido_Avellaneda', 'partido_Bahía Blanca',
       'partido_Balvanera', 'partido_Barracas', 'partido_Barrio Norte',
       'partido_Belgrano', 'partido_Boedo', 'partido_Caballito',
       'partido_Colegiales', 'partido_Córdoba', 'partido_Escobar',
       'partido_Esteban Echeverría', 'partido_Ezeiza', 'partido_Flores',
       'partido_Floresta', 'partido_General San Martín', 'partido_Ituzaingó',
       'partido_La Matanza', 'partido_La Plata', 'partido_Lanús',
       'partido_Lomas de Zamora', 'partido_Mar del Plata', 'partido_Monserrat',
       'partido_Moreno', 'partido_Morón', 'partido_Nuñez', 'partido_Palermo',
       'partido_Pilar', 'partido_Pinamar', 'partido_Punilla',
       'partido_Quilmes', 'partido_Recoleta', 'partido_Rosario',
       'partido_Saavedra', 'partido_San Cristobal', 'partido_San Fernando',
       'partido_San Isidro', 'partido_San Miguel', 'partido_San Telmo',
       'partido_Tigre', 'partido_Tres de Febrero', 'partido_Vicente López',
       'partido_Villa Carlos Paz', 'partido_Villa Crespo',
       'partido_Villa Devoto', 'partido_Villa Luro', 'partido_Villa Urquiza',
       'partido_Villa del Parque', 'prop_PH', 'prop_apartment', 'prop_house',
       'amb_1.0', 'amb_2.0', 'amb_3.0', 'amb_4.0', 'amb_5.0', 'amb_6.0',
       'amb_7.0'], axis = 1)
Y_train_superficie = X_train_superficie["precio_usd_por_m2"]
Y_test_superficie = X_test_superficie["precio_usd_por_m2"]

X_train_superficie.drop(["precio_usd_por_m2"], axis = 1, inplace = True)
X_test_superficie.drop(["precio_usd_por_m2"], axis = 1, inplace = True)
```

[36]:
```
lm_superficie = linear_model.LinearRegression()
# Fiteamos el modelo sobre los vectores X e Y.
model_superficie = lm_superficie.fit(X_train_superficie, Y_train_superficie)
```

[37]:
```
# Guardamos  las predicciones en un nuevo vector que llamaremos predictions.
predictions_superficie = lm_superficie.predict(X_test_superficie)

# Imprimimos el intercepto y los coeficientes como atributos del objeto␣
 ↪entrenado.
print ('Intercepto =', model_superficie.intercept_)
```

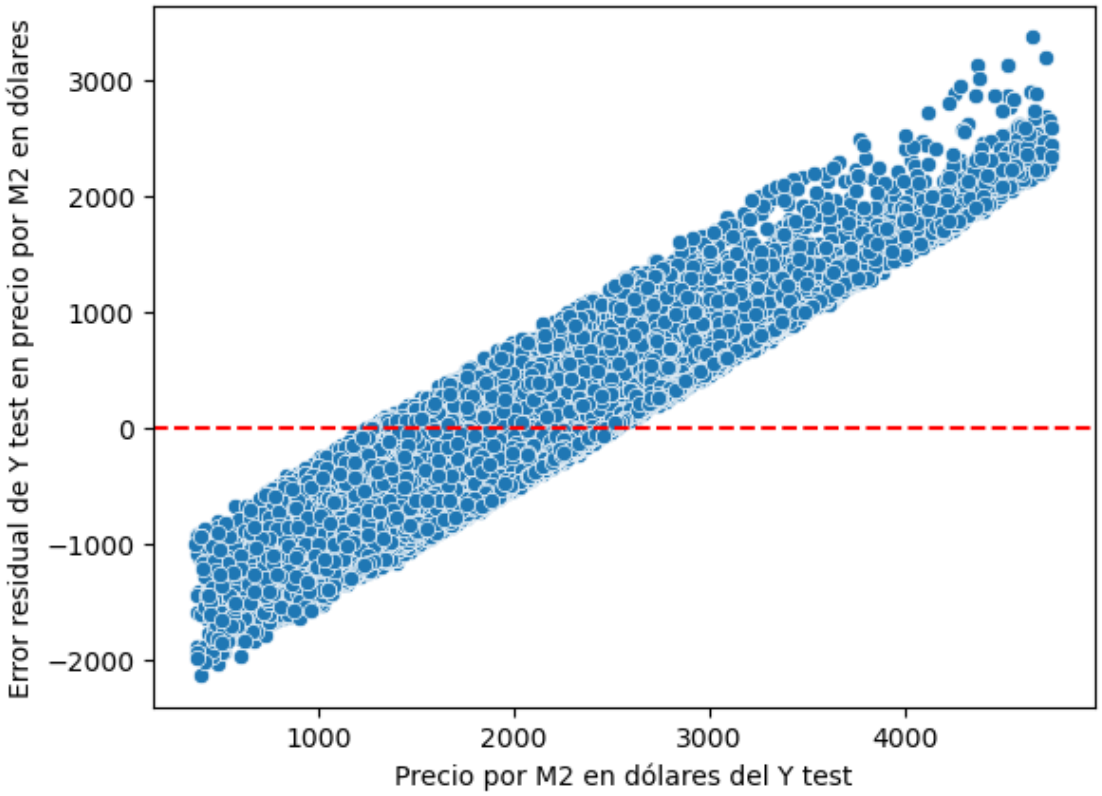
```
print ('RM =', model_superficie.coef_)

# Imprimos la metrica que mide la bondad de ajusto del modelo. En este caso el␣
 ↪R2.
print ('R2_train =', model_superficie.score(X_train_superficie,␣
 ↪Y_train_superficie))
```

```
Intercepto = 2625.7158102776366
RM = [-4.64378349]
R2_train = 0.08378760201532154
```

[38]:
```
test_residuals_superficie = Y_test_superficie - predictions_superficie
```

[39]:
```
sns.scatterplot(x = Y_test_superficie, y = test_residuals_superficie)
plt.axhline(y = 0, color = 'r', ls = "--")
plt.xlabel("Precio por M2 en dólares del Y test")
plt.ylabel("Error residual de Y test en precio por M2 en dólares")
```

[39]:
```
Text(0, 0.5, 'Error residual de Y test en precio por M2 en dólares')
```



[40]:
```
# Imprimos la metrica que mide la bondad de ajusto del modelo. En este caso el␣
 ↪R2.
R2_train_superficie = model_superficie.score(X_train_superficie,␣
 ↪Y_train_superficie)
R2_train_superficie
```

[40]: 0.08378760201532154

[41]:
```
# Error absoluto de la media
MAE_Test_superficie = mean_absolute_error(Y_test_superficie,␣
 ↪predictions_superficie)
MAE_Test_superficie
```

[41]: 630.7234348523389

[42]:
```
# Raiz cuadrada de la media del error
MSE_Test_superficie = mean_squared_error(Y_test_superficie,␣
 ↪predictions_superficie)
MSE_Test_superficie
```

[42]: 616548.9623586924

[43]:
```
RMSE_Test_superficie = np.sqrt(mean_squared_error(Y_test_superficie,␣
 ↪predictions_superficie))
RMSE_Test_superficie
```

[43]: 785.206318338494

Se crea un diccionario con los datos que se recopilan de ahora en adelante, así se comparan al final de la notebook 3

[44]:
```
revision_datos = [ {'subconjunto': 'Superficie Cubierta',
    'R2_train': R2_train_superficie.round(4),
    'MAE': MAE_Test_superficie.round(4),
    'MSE': MSE_Test_superficie.round(4),
    'RMSE': RMSE_Test_superficie.round(4)}]
```

4 5) Creación de modelo polinomial

[45]:
```
poly_train = PolynomialFeatures(2)
poly_features_train = poly_train.fit_transform(X_train_todo_dataset)
```

[46]:
```
poly_test = PolynomialFeatures(2)
poly_features_test = poly_test.fit_transform(X_test_todo_dataset)
```

[47]:
```

```

```
# X_train_todo_dataset, X_test_todo_dataset, Y_train_todo_dataset,␣
↪Y_test_todo_dataset = train_test_split (poly_features, Y_train_todo_dataset,␣
↪test_size = 0.3, random_state = 101)
```

[48]:
```
poly_model = linear_model.LinearRegression()
```

[49]:
```
poly_predictions = poly_model.fit(poly_features_train, Y_train_todo_dataset).
↪predict(poly_features_test)
```

[50]:
```
print ('R2_train =', poly_model.score(poly_features_train,␣
↪Y_train_todo_dataset))
```

```
R2_train = 0.5835339640788053
```

[51]:
```
poly_model.coef_
```

[51]:
```
array([ 1.69561191e+09, -2.43565914e+13,  8.25570450e+12, …,
        5.26876809e+10,  0.00000000e+00,  1.38475088e+11])
```

[52]:
```
MAE = mean_absolute_error(Y_test_todo_dataset, poly_predictions)
MAE
```

[52]:
```
20699761.61719425
```

[53]:
```
MSE = mean_squared_error(Y_test_todo_dataset, poly_predictions)
MSE
```

[53]:
```
1.5780722981377444e+18
```

[54]:
```
RMSE = np.sqrt(MSE)
RMSE
```

[54]:
```
1256213476.3398075
```

Exportamos el dataset para hacer gráficos y comparaciones

[55]:
```
data_con_dummies.to_csv('data_con_dummies.csv', index = False, sep=';')
```