09
22

INFORME PROPERATI

por DSConsultores

RESUMEN
de los objetivos

MODELOS
de Regresion

CONCLUSION
del trabajo realizado

RESUMEN

El presente informe tiene por objetivo mostrar de manera resumida todo el trabajo realizado para arribar a un modelo de machine learning que nos permita tasar una propiedad de manera automatica.
Todo el trabajo se realizo en Python en un entorno de Jupyter Lab.

Este trabajo esta compuesto por dos etapas:

En la **primer etapa** tuvimos como objetivos:

- Efectuar una limpieza del dataset provisto, diseñando estrategias para lidiar con los datos perdidos en ciertas variable.
- Realizar un análisis descriptivo de las principales variables
- Crear nuevas columnas a partir de las características dadas que puedan tener valor predictivo
- Arribar a un dataset depurado, ordenado y completo con datos certeros y robustos.

En esta **segunda etapa**, a partir del dataset depurado obtenido en el paso 1, se desarrollo un modelo de regresión lineal que nos permita predecir el precio en usd de una propiedad, para que la empresa pueda utilzarlo como tasador automático en las próximas propiedades que sean comercializadas.

A continuacion se muestra todo el trabajo realizado de esta segunda etapa y se adjunta el notebook con el codigo correspondiente.

INDICE

1

INTRODUCCION

En este segundo workshop del curso de Data Science de Digital House, nos enfocaremos en desarrollar un modelo de regresión que permita predecir el precio en usd de una propiedad, para que la inmobiliaria Properati pueda utilizarlo en un futuro como un tasador automático en las próximas propiedades que sean comercializadas.

Realizaremos distintos modelos para poder comparar los resultados obtenidos y arribar al óptimo.

Se plantearán diversas estrategias para abordar fallas y/o faltantes de información que perjudiquen al resultado del modelo, en vistas de lograr el objetivo mencionado anteriormente, todo esto lo haremos mediante la aplicación de los conocimientos adquiridos hasta el momento a lo largo del cursado.

¡Importante!

En todo el presente trabajo se utiliza de manera indistinta el nombre de las siguientes variables: Precio Total en USD = ARS_to_USD_corregido = ARS_to_USD

2

DATASET

El dataset contiene información sobre todas las propiedades georeferenciadas de la base de datos de la empresa.
La información que cada propiedad incluye es la siguiente:

operation: sell, rent, **property_type:** house, apartment, ph, **place_name**, **place_with_parent_names**, **country_name**, **state_name**, **geonames_id** (si está disponible), **lat-lon**, **price** (precio original del aviso), **currency:** ARS, USD, **price_aprox_local_currency:** ARS, **price_aprox_usd**, **surface_total_in_m2**, **surface_covered_in_m2**, **price_usd_per_m2**, **price_per_m2**, **floor:** (si corresponde), **rooms**, **expenses**, **properati_url**, **description**, **title**, **image_thumbnail**

Trabajamos con el dataset limpio obtenido luego de todo el proceso de etl realizado en el workshop 1, ademas solo nos enfocaremos en los registros de CABA y GBA (excluyendo los outliers), debido a que los mismos representan mas del 80% de los registros del dataset.

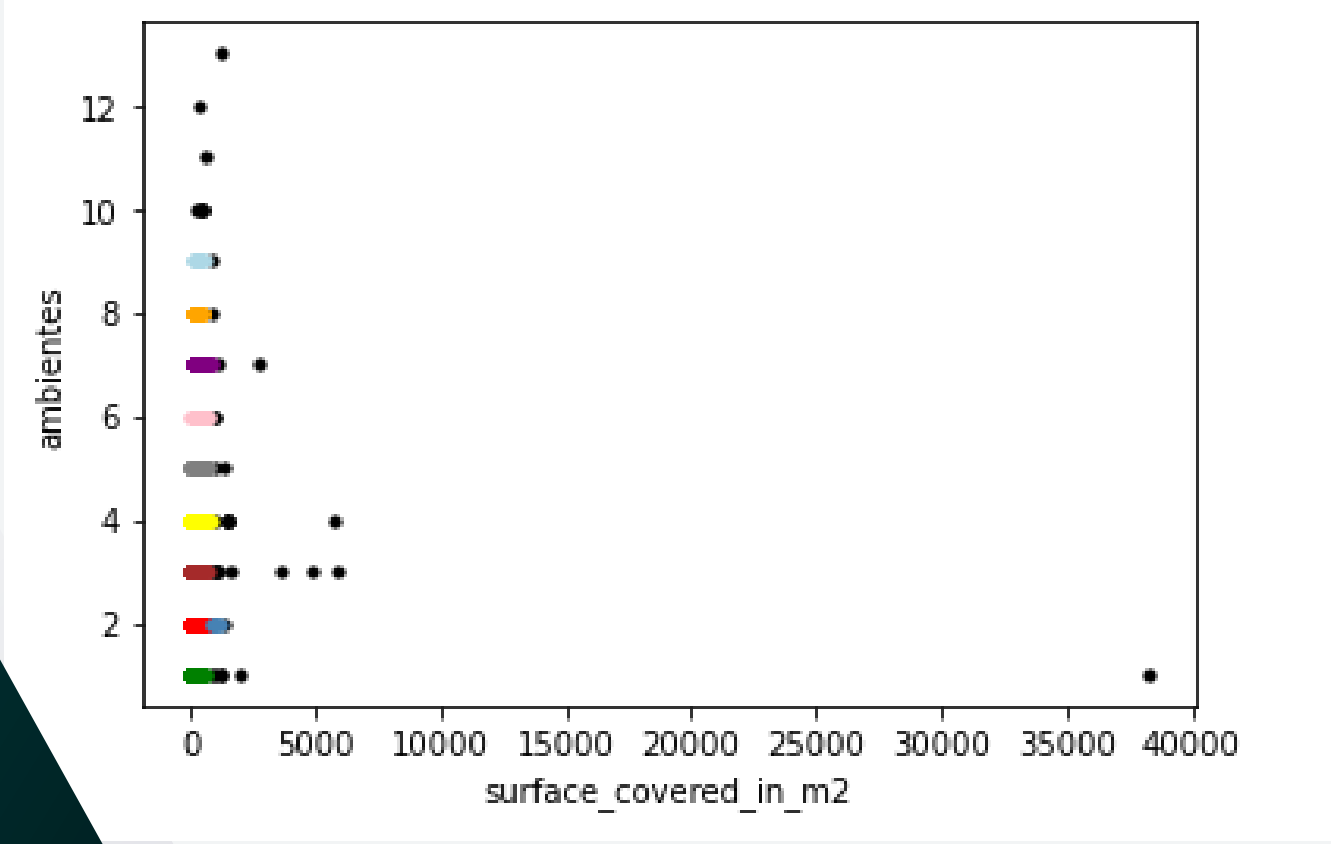
Realizaremos en esta etapa los pasos necesarios para: agregar las columnas que consideramos relevantes para nuestro analisis, quitar los nulos que perjudiquen al modelo y obtener el dataset lo más prolijo posible para luego entrenar nuestro modelo.

Lo primero que se realizo fue imputar los valores faltantes en la columna "ambientes" utilizando información de la columna "surface_covered_in_m2", pero para ello primero se completo esta última, utilizando información de la columna "surface_total_in_m2"

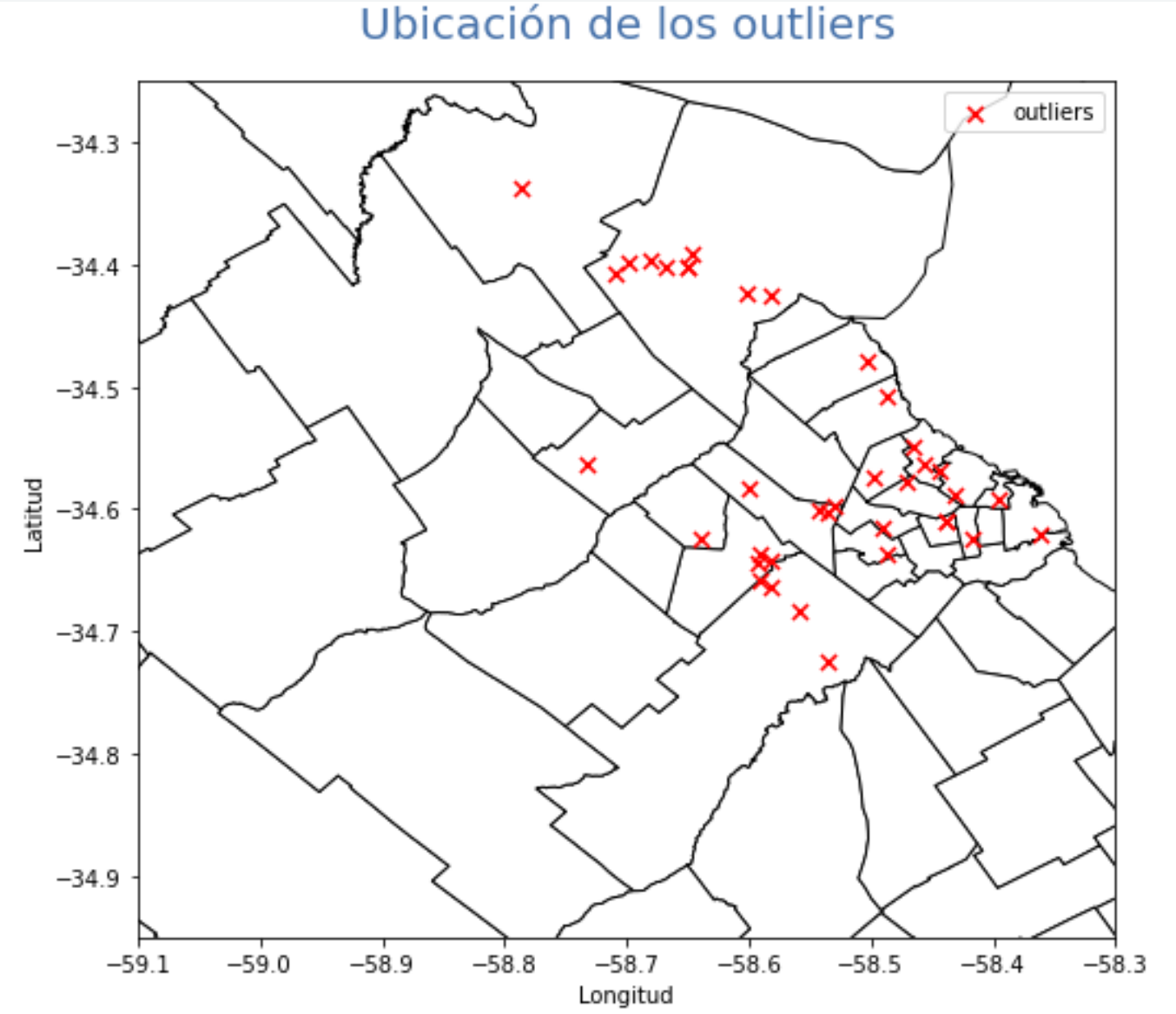
Del primer analisis se detectaron valores probablemente erróneos, como departamentos con 85 ambientes. Sin embargo, observamos que para todos los tipos de propiedad, la mediana se encontraba cerca de la media, ligeramente por debajo (seguramente debido a la presencia de ese tipo de outliers). La diferencia promedio entre estas dos magnitudes, para todos los tipos de propiedad, es de aprox. 15%.
Tomamos la **Mediana** como valor de referencia a la hora de imputar, ya que además se trata de un número entero.

En el analisis de la columna "superficie cubierta"(_surface_covered_in_m2_), la diferencia entre los dos estadísticos es mayor, lo que indica mayor presencia de outliers, sobre todo para el tipo de propiedad 'stores'.
Detectamos que, en promedio, en los primeros 3 tipos de propiedad, hay una diferencia de aprox. 24% (la media por encima de la mediana), pero en el caso de stores, esta diferencia asciende a 171%, elevando el promedio total significativamente.
Se tuvo todo esto en cuenta a la hora de tratar de corregir posibles outliers y luego efectuar la imputación.

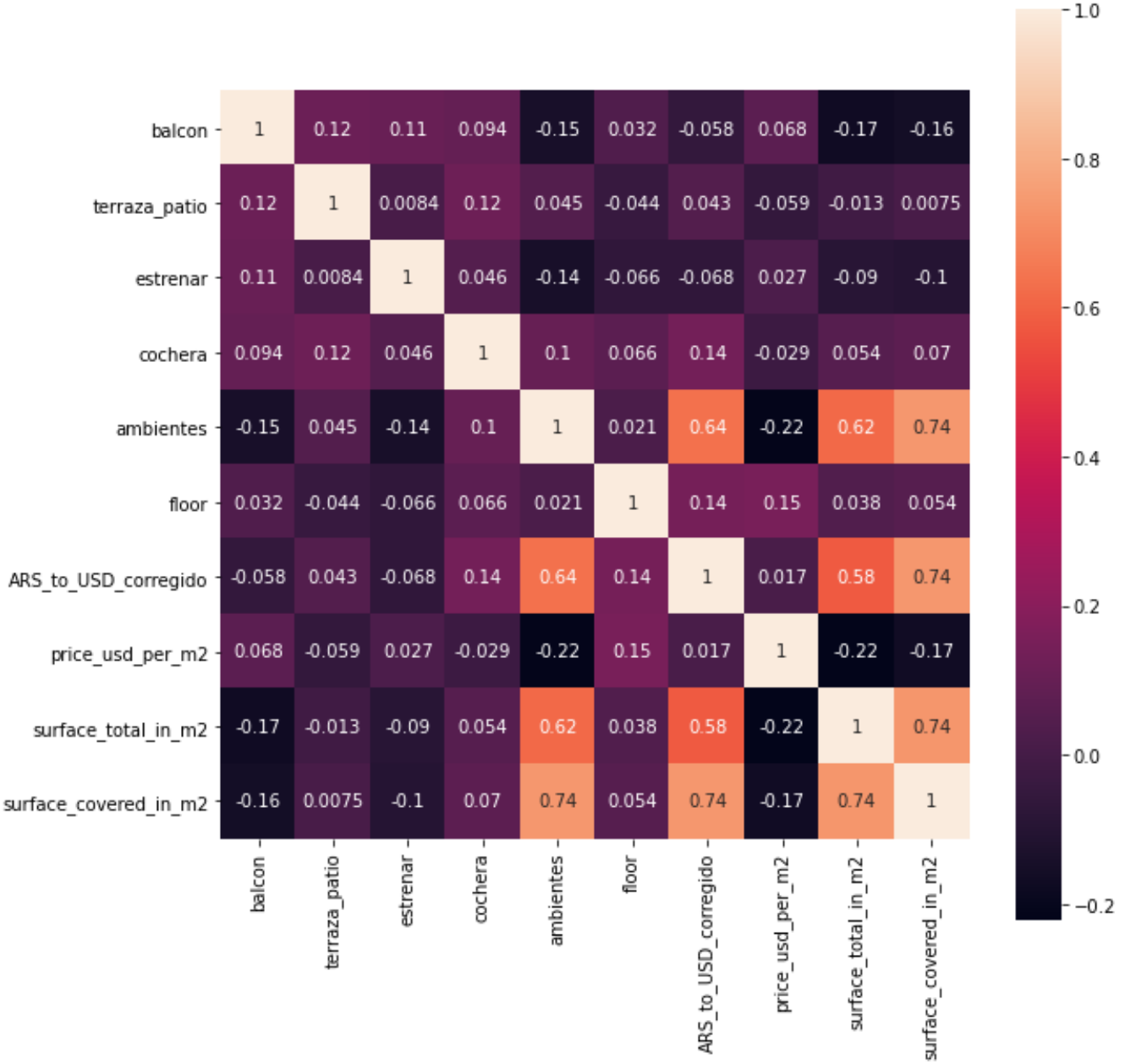
Paso siguiente se utilizo **DBScan** para agrupar por ambientes y superficie cubierta, teniendo en cuenta la ubicación de las diferentes propiedad.



Con esto pudimos detectar y eliminar outliers.



Luego analizamos cómo se relacionan las columnas entre sí con la Matriz de Correlación:



Del gráfico anterior, observamos que hay una alta correlación entre el Precio Total en USD ('ARS_to_USD_corregido') y la Superficie Total ('surface_total_in_m2'), así que vamos a usar a la Superficie Total como feature para predecir el Precio Total en USD con el modelo de Regresión Lineal Simple. En segundo lugar, vemos que el Precio Total está bastante correlacionado también con la cantidad de Ambientes.

3 FEATURE ENGINEERING

(Modelado)

1) Vamos a **crear variables Dummies** para las siguientes **variables Categóricas**:
- property_type ;CABA_or_GBA ; barrio (anteriormente llamada 'place_name')

```
#Creamos variables dummies de las variables categóricas:

property_type_dummies=pd.get_dummies(df_properatti.property_type, prefix= 'property_type', drop_first=True)

CABA_or_GBA_dummies=pd.get_dummies(df_properatti.CABA_or_GBA, drop_first=True) #quité el pefix porque era medio confuso CABA_or_GBA_GBA

barrio_dummies=pd.get_dummies(df_properatti.barrio, drop_first=True) #quite el prefix porque necesito los nombres de los barrios limpios
```

```
#Concatenamos el Dataframe original y los Dummy Dataframes (axis=0 significa filas, axis=1 significa columnas):
df_properatti=pd.concat([df_properatti,property_type_dummies], axis=1)
df_properatti=pd.concat([df_properatti,CABA_or_GBA_dummies], axis=1)
df_properatti=pd.concat([df_properatti,barrio_dummies], axis=1)
```

2) **Estandarizamos** las **variables numéricas**:
- surface_total_in_m2 ; surface_covered_in_m2 ; ambientes

```
#usamos StandarScaler como para escaloar los valores ya que MaxMin puede causar problemas con los outliers que puedan haber
numericals = ['surface_total_in_m2','surface_covered_in_m2', 'ambientes']#,'sup_total_2', 'ambientes_2']#,'estrenar','cochera']

X = df_properatti[numericals]
lm = linear_model.LinearRegression()
scaler = StandardScaler()                          # va a escalar los valores entre 0 y 1, y lo instanciamos con el nombre del objeto "scaler".
X_std = scaler.fit_transform(X)                                  # Vamos a decirle a scaler que fittee, que aprenda de X los valores
#model = lm.fit(X_std, y)
std_df = pd.DataFrame(X_std)
std_df.columns = [i + '_std' for i in numericals]      # renombramos a las columnas con un i_std, para no confundirnos con las columnas que tienen igual nombre en el dataset original
std_df.head(3)
```

	surface_total_in_m2_std	surface_covered_in_m2_std	ambientes_std
0	-0.454268	-0.584000	-0.673080
1	-0.454268	-0.474619	-0.673080
2	-0.491078	-0.584000	-1.169155

3) **Construimos nuestra matriz de features** como la **Concatenación de dummies_df y el std_df** (añadimos también las variables de 'estrenar' y 'cochera':
- surface_total_in_m2 ; surface_covered_in_m2 ; ambientes

Entonces X, que es el dataset de features que usamos para el entrenamiento, va a ser concatenar todas las variables dummies que construimos antes (dummies_df) con todas las variables numéricas estandarizadas (std_df). Este X es básicamente nuestro nuevo dataset que vamos a usar para entrenar el modelo de Regresión Lineal:

```
#valores que usaremos de ahora en adelante como X e y
X = pd.concat([dummies_df, std_df,df_properatti[['estrenar', 'cochera']].astype(int)], axis=1)
y = df_properatti.ARS_to_USD_corregido

X.tail(3)
```

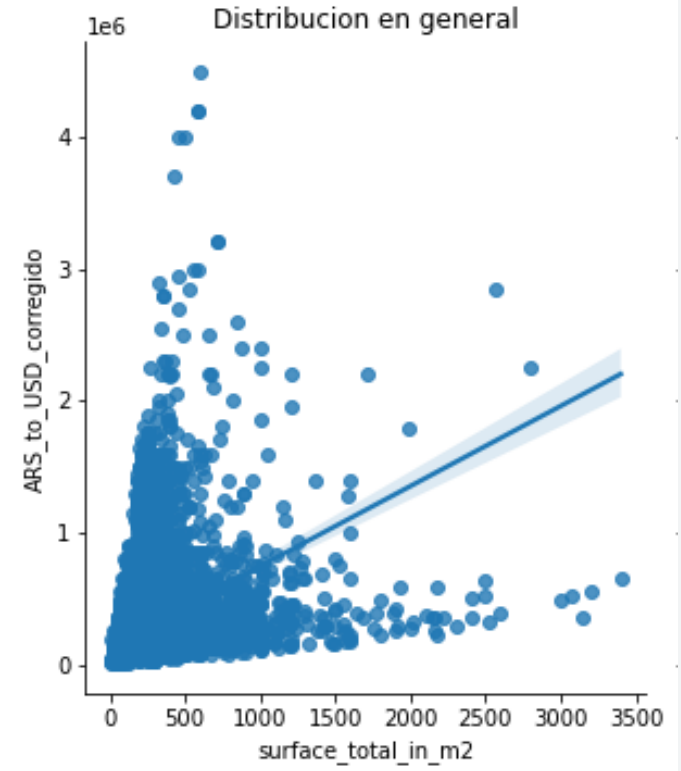
	property_type_apartment	property_type_house	property_type_store	GBA	Acacias Blancas	Acassuso	Agronomía	Albanueva Barrio Cerrado	Aldo Bonzi	Almagro	...	Villa del Parque	Village Golf & Tennis Country Club	Virrey del Pino	Virreyes	Zelaya	surface_total_in_m2_std	surface_covered_in_m2_std	ambientes_std	estrenar	cochera
43241	0	1	0	1	0	0	0	0	0	0	...	0	0	0	0	0	0.675775	1.166088	1.807292	1	1
43242	1	0	0	1	0	0	0	0	0	0	...	0	0	0	0	0	-0.303351	-0.299611	0.319069	0	0
43243	0	1	0	1	0	0	0	0	0	0	...	0	0	0	0	0	0.668413	1.749450	2.303367	0	1

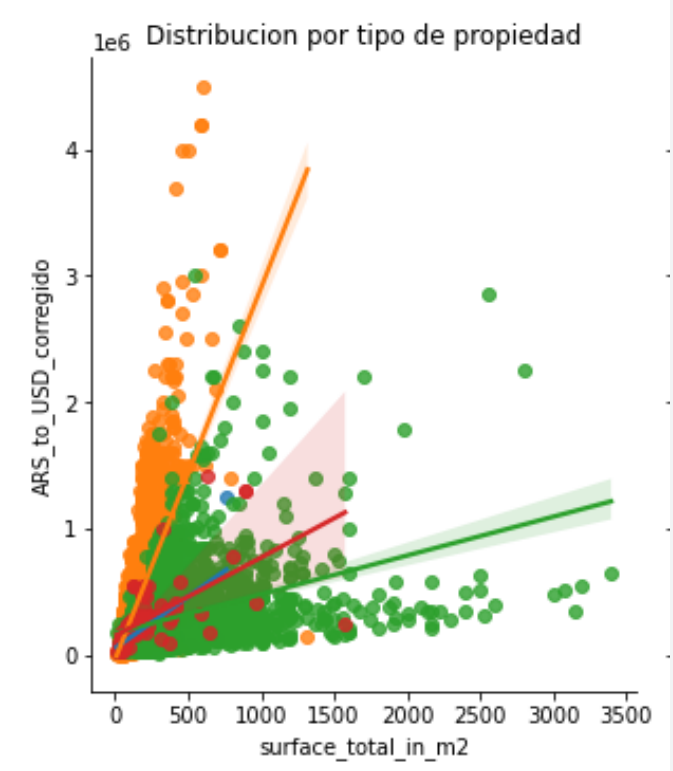
3 rows × 321 columns

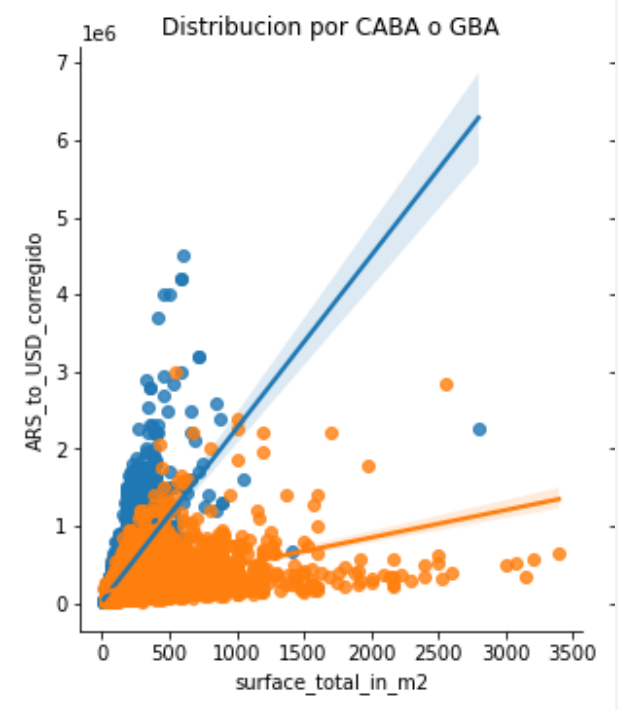
Y la variable Target "y" es el Precio Total en USD ("ARS_to_USD_corregido"), y no está estandarizada ni escalada ni nada, la vamos a usar así como la leímos de los datos crudos:

4 REGRESION LINEAL SIMPLE

Empezamos analizando una Regresión Lineal Simple usando como feature (Variable independiente "X") o variable Predictora a la "surface_total_in_m2", que era la variable que más estaba correlacionada con el Precio Total en USD ("ARS_to_USD_corregido")
Hacemos el Scatterplot entre 'surface_total_in_m2' y 'ARS_to_USD_corregido' para ver a priori qué correlación hay, analizando luego por tipo de propiedad y distribucion por CABA o GBA

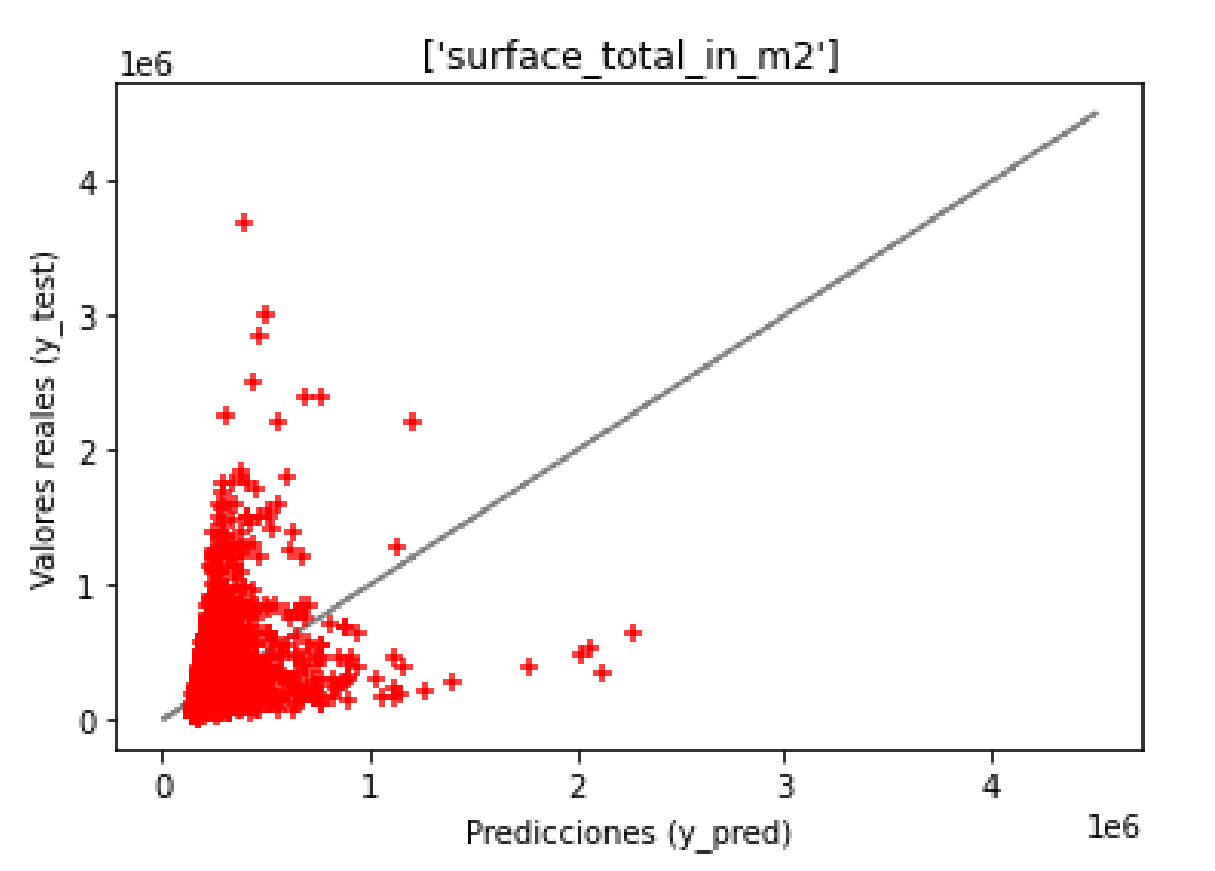






Vemos que es una dispersión que no se puede representar con una recta de una manera que tenga precisión, ya que cualquier recta que se trace, por más de que minimice los errores, siempre va a haber muchos puntos por encima de la recta y muchos puntos por debajo de la recta. Por lo que ya sabemos de entrada que este modelo de Regresión lineal simple no va a lograr un nivel aceptable de predicción.

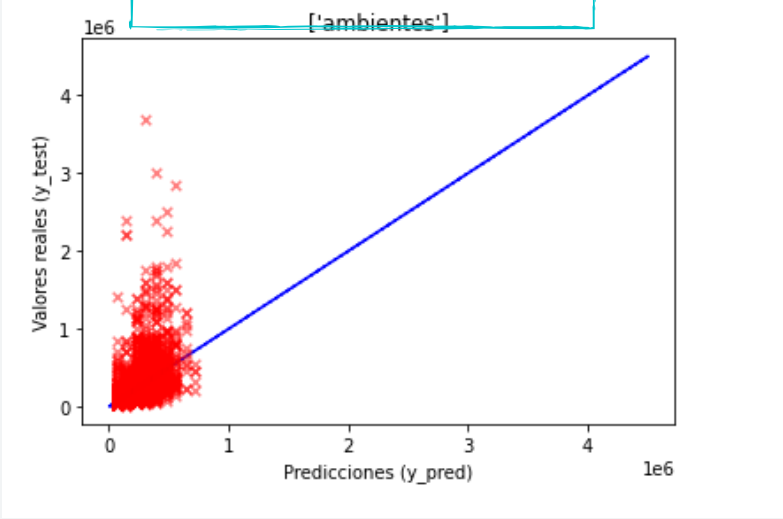
Ahora, veamos gráficamente cómo se comparan los valores predichos por el modelo (y_pred) con los valores reales (y_test):



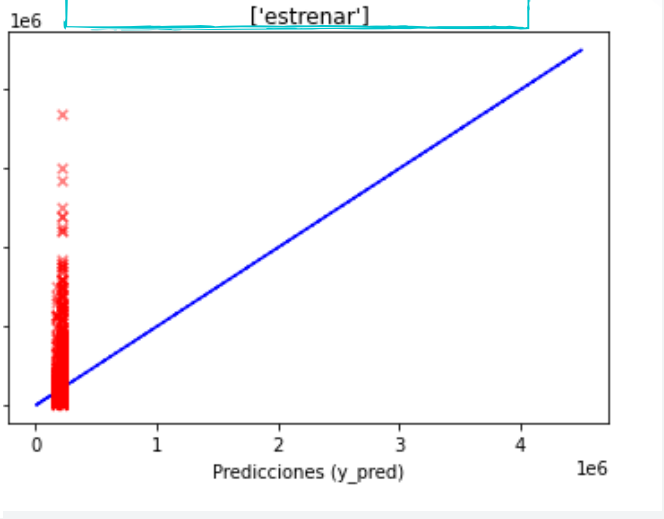
R2_test = 0.3051

Realizamos distintas regresiones lineales para comparar diferentes ensambles de features y ver como se relaciona la variable target Precio Total en USD ("ARS_to_USD_corregido")

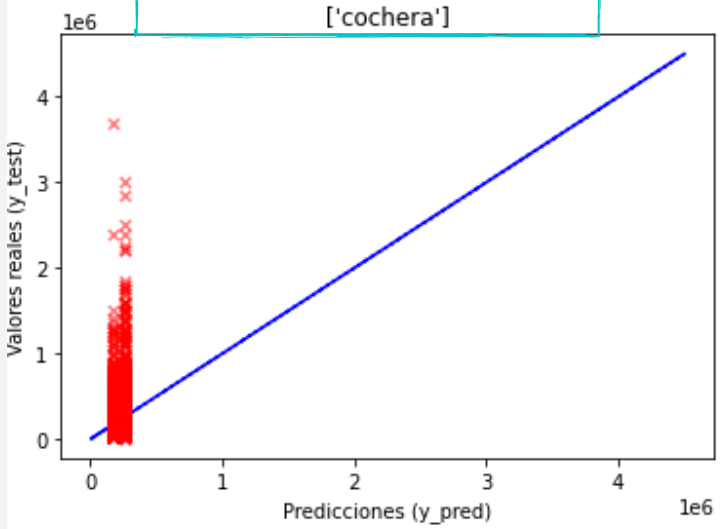
R2_test = 0.4573



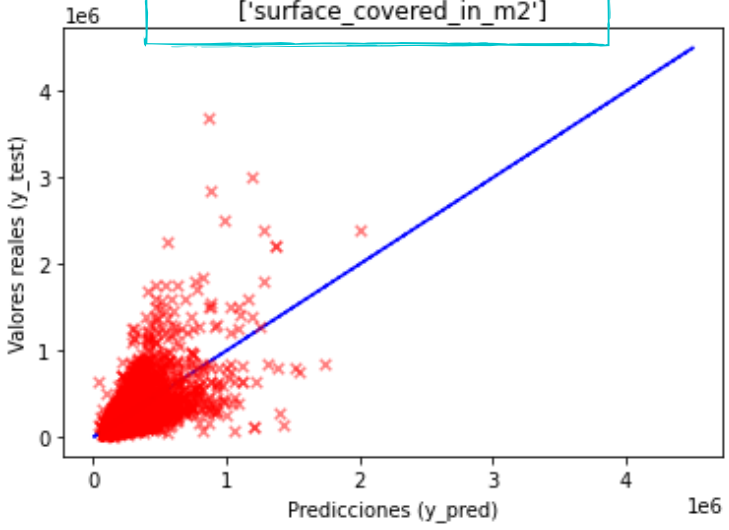
R2_test = 0.0049



R2_test = 0.0207



R2_test = 0.0207



Comparamos resultados:

```
['cochera']
Intercept:  229221.012
Coeficients:  [87088.017]
[('cochera', 87088.017)]
y_test_sample: [310000. 380000.  60000. 185000. 128300.]
y_pred_sample: [316309 316309 229221 316309 229221]
MAE:  174862.941
MSE:  85726454336.423
RMSE:  292790.803
R2_train:  0.019
R2_test:  0.021
```

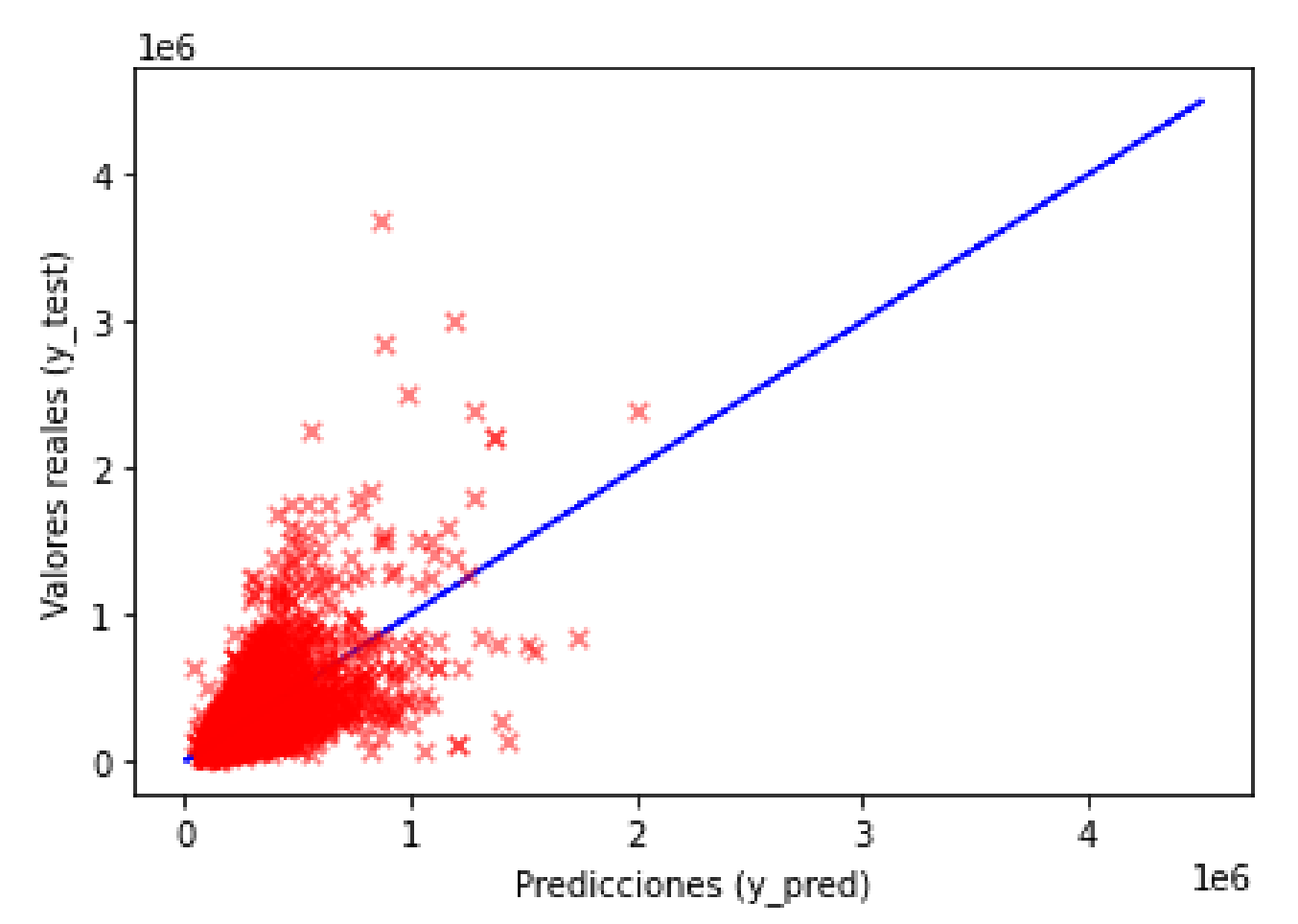
```
['estrenar']
Intercept:  271333.673
Coeficients:  [-62224.559]
[('estrenar', -62224.559)]
y_test_sample: [310000. 380000.  60000. 185000. 128300.]
y_pred_sample: [271333 271333 271333 271333 271333]
MAE:  176071.059
MSE:  87113875249.956
RMSE:  295150.598
R2_train:  0.004
R2_test:  0.005
```

```
['ambientes']
Intercept:  -57642.763
Coeficients:  [95967.723]
[('ambientes', 95967.723)]
y_test_sample: [310000. 380000.  60000. 185000. 128300.]
y_pred_sample: [230260 230260 134292 134292 230260]
MAE:  118063.264
MSE:  47509980141.042
RMSE:  217967.842
R2_train:  0.389
R2_test:  0.457
```

```
['surface_covered_in_m2']
Intercept:  65542.983
Coeficients:  [1654.274]
[('surface_covered_in_m2', 1654.274)]
y_test_sample: [310000. 380000.  60000. 185000. 128300.]
y_pred_sample: [230970 217736 131713 187959 150456]
MAE:  100156.406
MSE:  39686265970.361
RMSE:  199214.121
R2_train:  0.548
R2_test:  0.547
```

```
['surface_total_in_m2']
Intercept:  146866.299
Coeficients:  [652.835]
[('surface_total_in_m2', 652.835)]
y_test_sample: [310000. 380000.  60000. 185000. 128300.]
y_pred_sample: [216066 212149 172979 195176 185383]
MAE:  131846.186
MSE:  60833822827.62
RMSE:  246645.135
R2_train:  0.34
R2_test:  0.305
```

Graficamos el modelo:



Podemos observar que el modelo de Regresion Lineal Simple es demasiado simple para poder obtener un *insight* lo suficientemente convincente, por lo cual vamos a aplicar dos modelos que incluyen técnicas de Regularizacion (Ridge y Lasso) para mejorar el R2 del modelo

5 REGRESION LINEAL MULTIPLE

5.1) Regresion Lineal Multiple SIN Regularizacion

Empezamos realizando una Regresión Lineal Múltiple sin Estandarización de Features Numéricas ni Creación de variables Dummies, para ver cómo nos da el Modelo. Usaremos 2 variables predictoras en nuestro modelo de Regresión Lineal Múltiple para empezar: a "surface_total_in_m2" y a "ambientes" y obtenemos los siguientes valores:

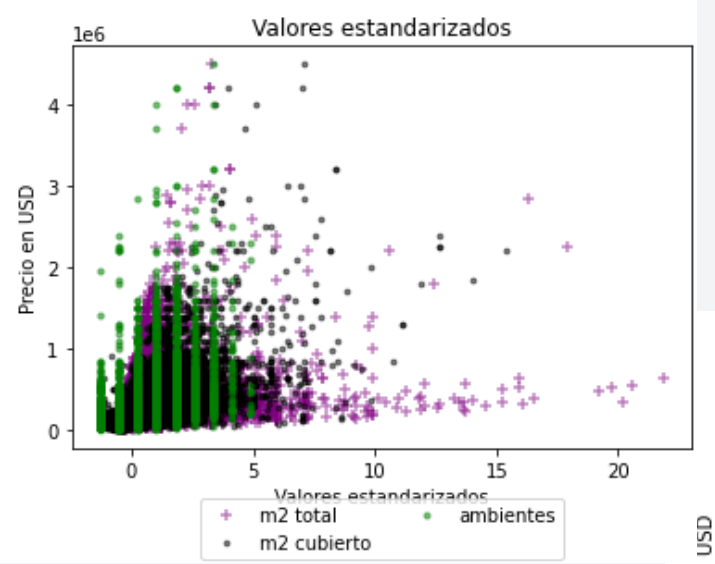
```
['surface_total_in_m2', 'surface_covered_in_m2', 'ambientes', 'estrenar', 'cochera']
Intercept:  -8518.652
Coeficients:  [ -169.25   1968.761 18255.51  19101.703 42735.749]
[('surface_total_in_m2', -169.25), ('surface_covered_in_m2', 1968.761), ('ambientes', 18255.51), ('estrenar', 19101.703), ('cochera', 42735.74
9)]
y_test_sample: [390000.         780000.          90000.         250000.
 170000.         140000.          55000.         133070.41908582
  33200.          75000.        ]
y_pred_sample: [140176 881809  99972 188594 153033 329744 128072  64675  63722  86648]
MAE:  75881.189
MSE:  20574231673.496
RMSE:  143437.205
```

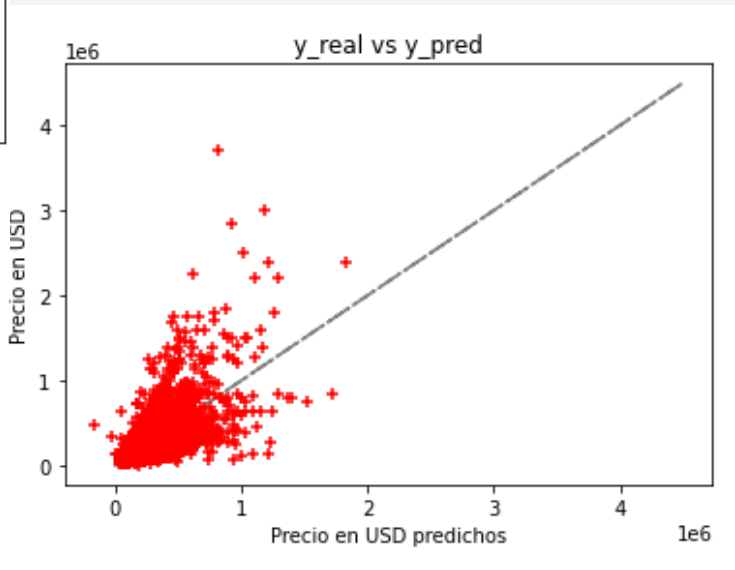
R2_test = 0.5700

5.2) Regresion Lineal Multiple CON Regularizacion

Estandarizacion de features numericas:
En este paso primero vamos a Normalizar (Estandarizar) las Features Numéricas: Con esto llevamos a las variables numéricas a la misma escala y le damos el mismo peso a todas. Las variables dummies no es necesario que sean normalizadas ya que son variables dicotomicas (0 o 1). Recordar que en la lista numericals vamos a poner a todas las variables numéricas, pero no a la variable Target (precio). Vamos a estandarizar las variables predictoras pero no vamos a estandarizar a la variable Target.

Observamos los valores estandarizados y los resultados del modelo:





OLS Regression Results

Dep. Variable:	ARS_to_USD_corregido	**R-squared:**	0.708
Model:	OLS	**Adj. R-squared:**	0.705
Method:	Least Squares	**F-statistic:**	283.4
Date:	Tue, 13 Sep 2022	**Prob (F-statistic):**	0.00
Time:	12:55:56	**Log-Likelihood:**	-3.9759e+05
No. Observations:	30477	**AIC:**	7.957e+05
Df Residuals:	30218	**BIC:**	7.979e+05
Df Model:	258		
Covariance Type:	nonrobust		

6 REGRESION LINEAL RIDGE

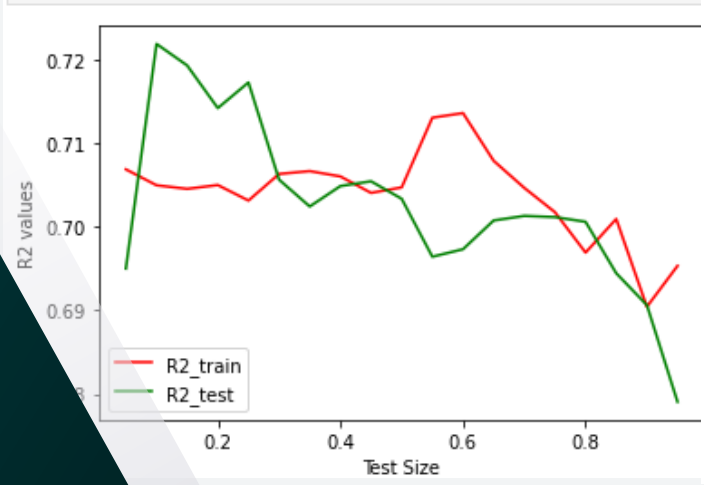
En una primer paso Instanciamos el modelo de Ridge, obteniendo un Score del modelo Ridge: 0.7031262057430505. Detectamos el alpha optimo = 1

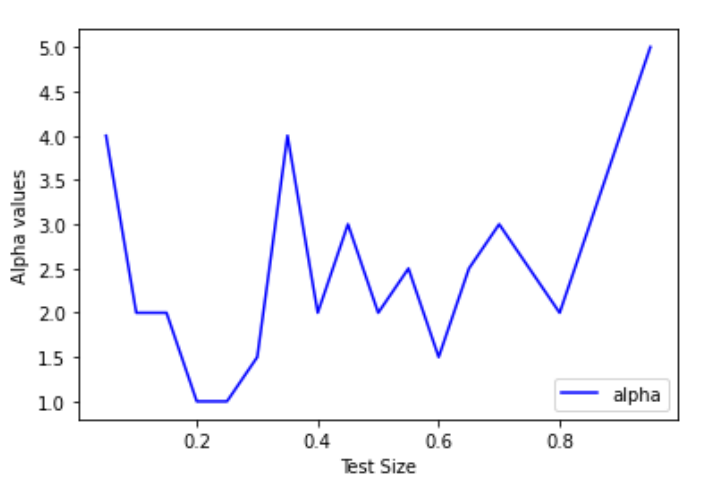
Luego Analizamos como reacciona R2 y alpha a los cambios en el % de test_size y en el n de folds, en la regularización de Ridge

Resultados obtenidos al analizar los cambios de R2 y alpha cuando variamos el % de test usado para testear el modelo

	test_size	R2_train	R2_test	best_alpha
0	0.05	0.706858	0.694998	4.0
1	0.10	0.704953	0.721852	2.0
2	0.15	0.704526	0.719273	2.0
3	0.20	0.704980	0.714177	1.0
4	0.25	0.703126	0.717245	1.0
5	0.30	0.706308	0.705618	1.5
6	0.35	0.706646	0.702417	4.0
7	0.40	0.706010	0.704861	2.0
8	0.45	0.704042	0.705429	3.0
9	0.50	0.704699	0.703338	2.0
10	0.55	0.713030	0.696412	2.5
11	0.60	0.713581	0.697308	1.5
12	0.65	0.707879	0.700745	2.5
13	0.70	0.704642	0.701294	3.0
14	0.75	0.701738	0.701150	2.5
15	0.80	0.696907	0.700588	2.0
16	0.85	0.700926	0.694440	3.0
17	0.90	0.690430	0.690633	4.0
18	0.95	0.695320	0.679072	5.0

Graficamos los R2 y los alpha:

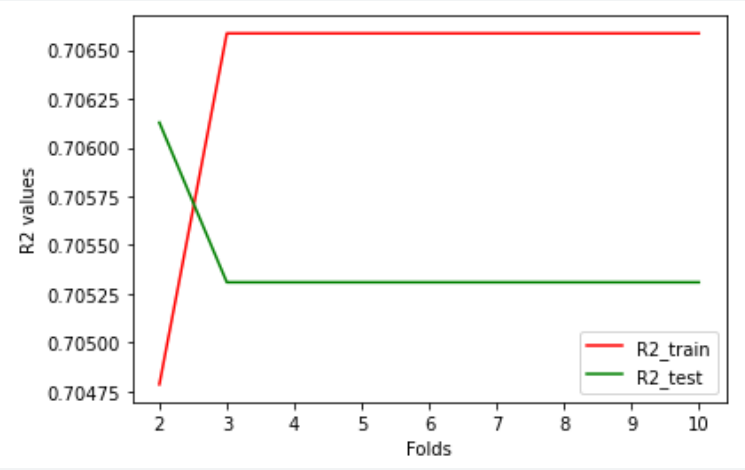


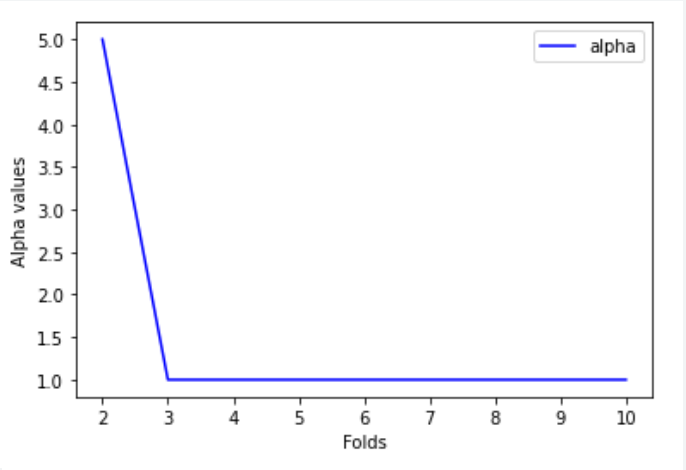


Podemos observar para este modelo que el valor optimo de R2 para test seria el 0.72

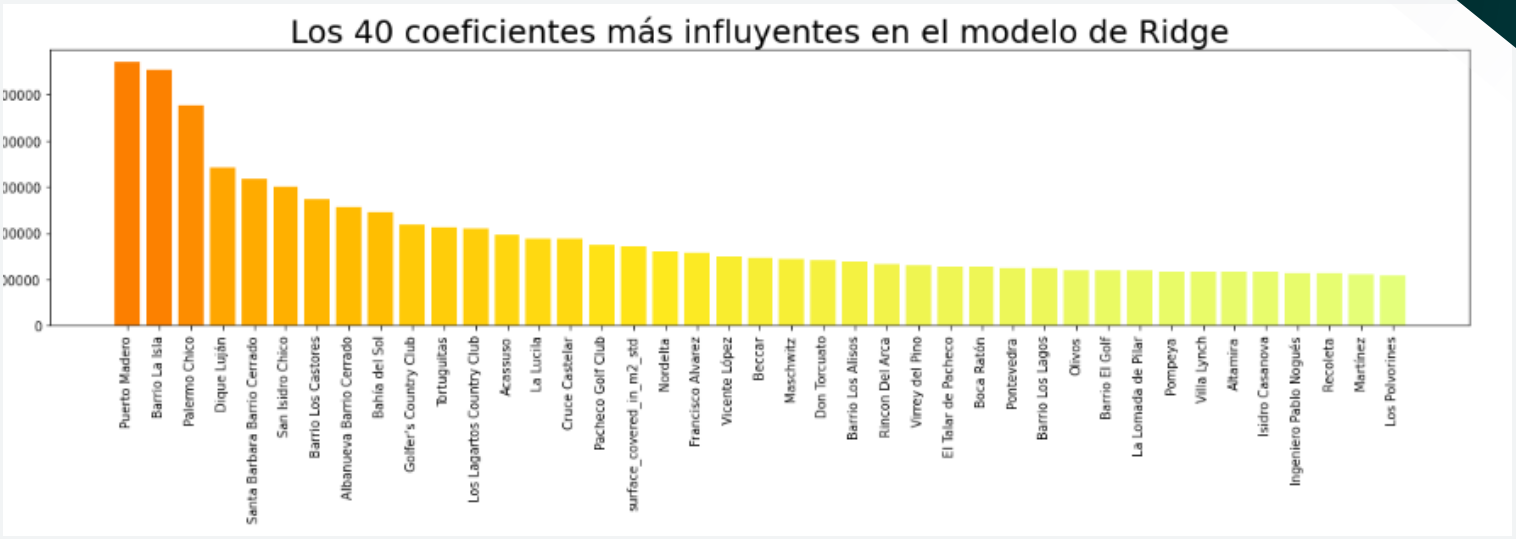
Resultados obtenidos al analizar los cambios de R2 y alpha cuando variamos el n de folds

	Folds	R2_train	R2_test	best_alpha
0	2	0.704787	0.706128	5.0
1	3	0.706584	0.705310	1.0
2	4	0.706584	0.705310	1.0
3	5	0.706584	0.705310	1.0
4	6	0.706584	0.705310	1.0
5	7	0.706584	0.705310	1.0
6	8	0.706584	0.705310	1.0
7	9	0.706584	0.705310	1.0
8	10	0.706584	0.705310	1.0

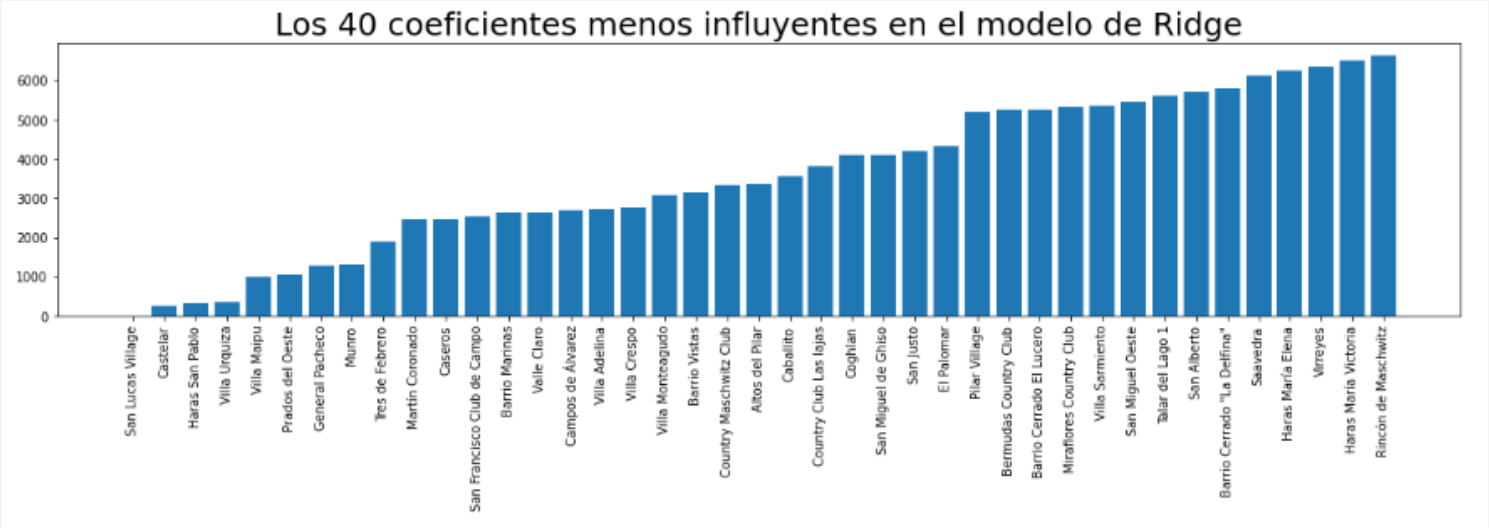




Podemos observar en este caso que el valor optimo de R2 para test seria el 0.7061 para 2 folds y un alpha = 5



En el gráfico en la parte superior de la hoja podemos observar los 40 coeficientes más influyentes en el modelo de Ridge (los de mayor Coeficiente).
Por otro lado, en la tabla inferior podemos observar a las 40 variables menos influyentes.

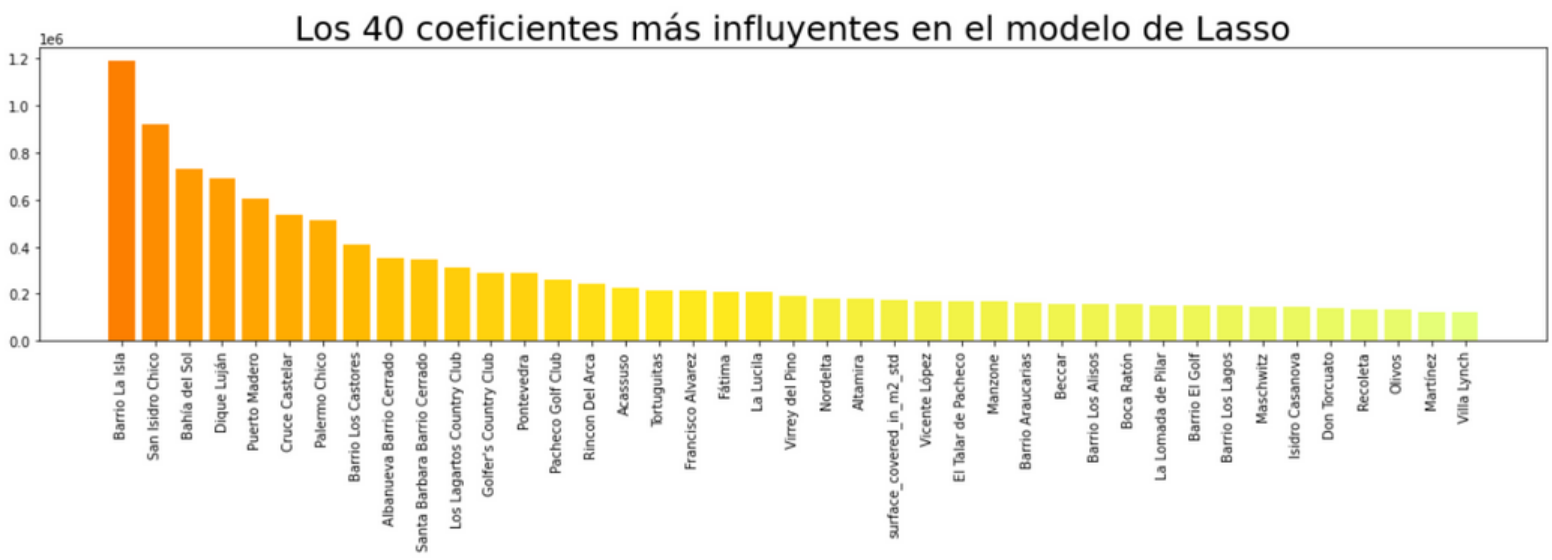


R2_test = 0.7183

por DSConsultores

7 REGRESION LINEAL LASSO

En una primer paso Instanciamos el modelo de Lasso, obteniendo un Score del modelo de: 0.705967019473495.
Detectamos el alpha optimo = 9.091818181818182



En el gráfico en la parte superior de la hoja podemos observar los 40 coeficientes más influyentes en el modelo de Lasso (los de mayor Coeficiente).

Por otro lado, en la tabla de la derecha podemos observar a las 30 variables menos influyentes (que Lasso las elimina del modelo haciendo que tomen valor = 0):

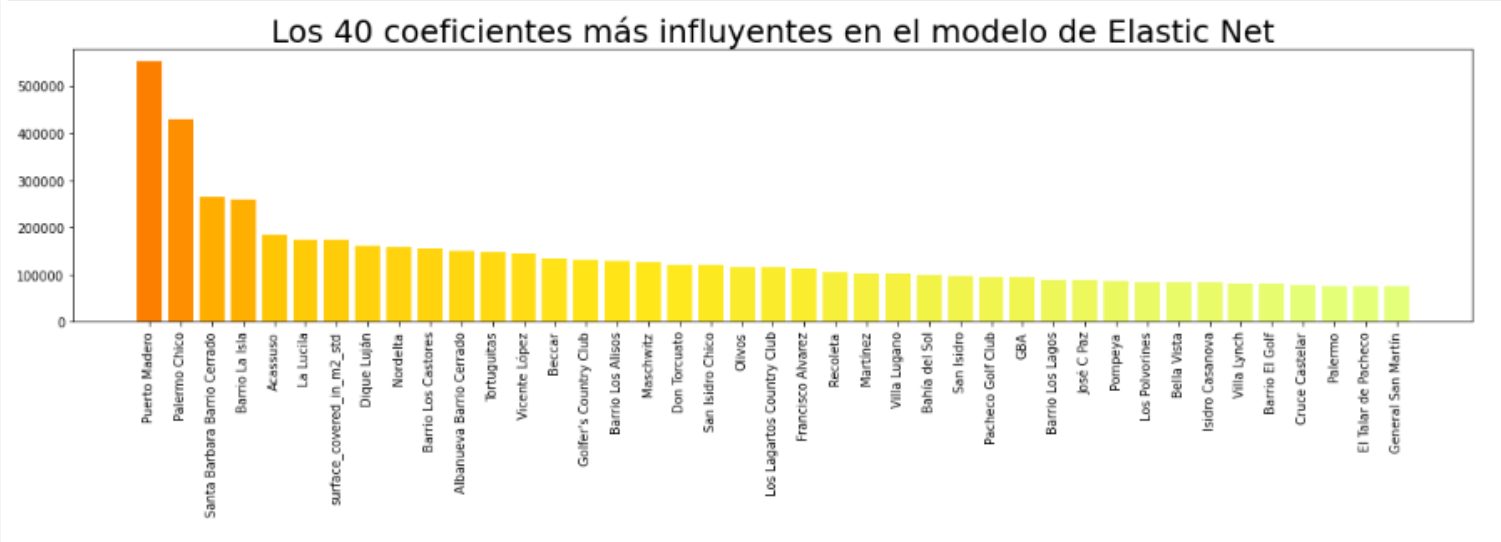
	Variable	Coefficient
6	Agronomía	-0.0
8	Aldo Bonzi	0.0
11	Altos de Manzanares 1 y 2	-0.0
12	Altos de Matheu	-0.0
13	Altos del Golf	-0.0
14	Altos del Pilar	-0.0
15	Armenia Country Club	-0.0
21	Barrancas de Santa María	0.0
24	Barrio Barrancas del Lago	-0.0
27	Barrio Cerrado "Buenos Aires Village"	0.0
28	Barrio Cerrado "La Cautiva del Pilar"	0.0
29	Barrio Cerrado "La Delfina"	0.0
30	Barrio Cerrado "La Montura"	-0.0
31	Barrio Cerrado "La Tranquera"	0.0
32	Barrio Cerrado "Los Alcanfores"	-0.0
34	Barrio Cerrado "Los Senderos"	-0.0
35	Barrio Cerrado "Soles de Pilar"	-0.0
36	Barrio Cerrado "Tres Horquetas"	-0.0
37	Barrio Cerrado El Lucero	0.0
42	Barrio La Cuesta	0.0
48	Barrio Marinas	0.0
50	Barrio Parque General San Martin	-0.0
51	Barrio Parque San Martin	0.0
53	Barrio Privado El Recodo S.A.	-0.0
54	Barrio San Agustin	0.0
56	Barrio San Matías	-0.0
60	Barrio Vistas	0.0
67	Benavidez Greens	0.0
68	Bermudas Country Club	0.0
70	Boat Center Barrio Cerrado	0.0

R2_test = 0.7175

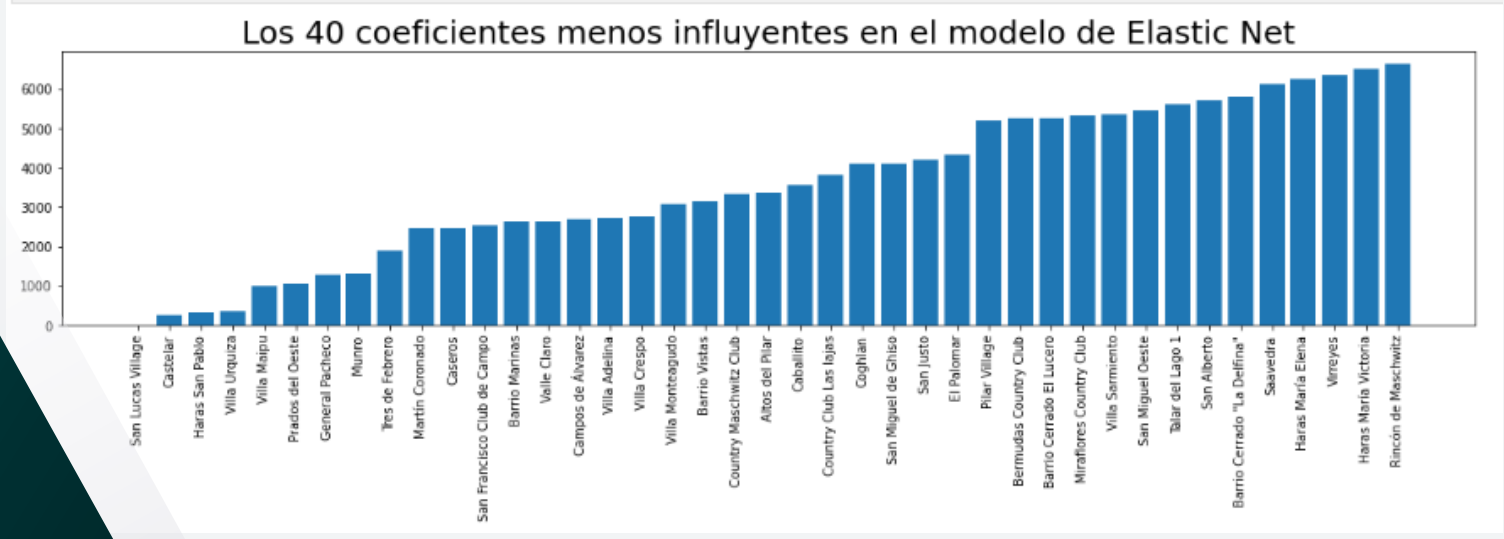
8

REGRESION ELASTIC NET

En una primer paso Instanciamos el modelo Elastic Net, obteniendo un Score del modelo de: 0.7058350003766158
Detectamos el alpha optimo = 10



En el gráfico en la parte superior de la hoja podemos observar los 40 coeficientes más influyentes en el modelo de Elastic Net (los de mayor Coeficiente).
Por otro lado, en la tabla inferior podemos observar a las 40 variables menos influyentes.



R2_test = 0.7165

por DSConsultores

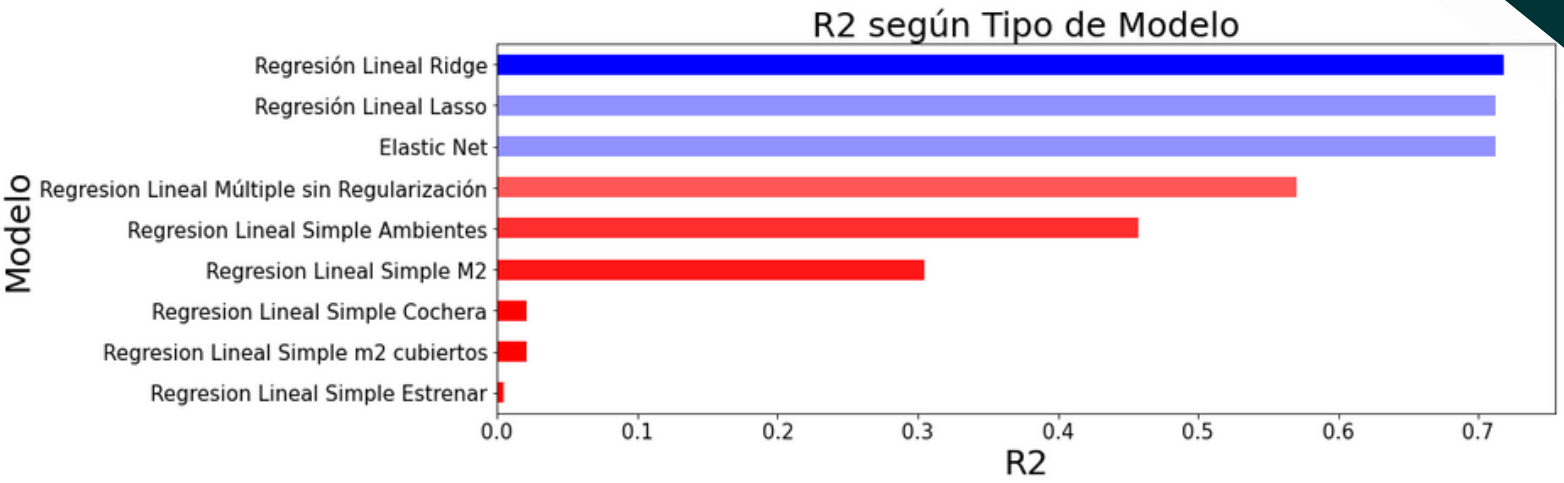
9 CONCLUSIONES

En la tabla siguiente vemos un resumen de los resultados obtenidos mediante los distintos modelos de Regresión Lineal:

COMPARACIÓN ENTRE MODELOS:

	Modelo	Alcance del modelo	Variable Objetivo	Cantidad de Observaciones	R2_train	R2_test	Intercepto	Alpha del modelo	l1_ratio
0	Regresion Lineal Simple M2	Buenos Aires	ARS_to_USD_corregido	43244	0.339744	0.305088	146866.298735	NaN	NaN
1	Regresion Lineal Simple Ambientes	Buenos Aires	ARS_to_USD_corregido	43244	0.389220	0.457288	-57642.763197	NaN	NaN
2	Regresion Lineal Simple Estrenar	Buenos Aires	ARS_to_USD_corregido	43244	0.004429	0.004888	271333.673305	NaN	NaN
3	Regresion Lineal Simple Cochera	Buenos Aires	ARS_to_USD_corregido	43244	0.019051	0.020737	229221.012187	NaN	NaN
4	Regresion Lineal Simple m2 cubiertos	Buenos Aires	ARS_to_USD_corregido	43244	0.547569	0.546659	65542.982970	NaN	NaN
5	Regresión Lineal Ridge	Buenos Aires	ARS_to_USD_corregido	43244	0.693722	0.718289	242429.987428	5.000000	NaN
6	Regresión Lineal Lasso	Buenos Aires	ARS_to_USD_corregido	43244	0.695829	0.717474	220693.667289	7.071636	NaN
7	Elastic Net	Buenos Aires	ARS_to_USD_corregido	43244	0.694596	0.716466	240787.737595	0.001000	0.9

Modelo	R2_train	R2_test
Regresión Lineal Ridge	0.6937	0.7183
Regresión Lineal Lasso	0.6958	0.7175
Elastic Net	0.6946	0.7165
Regresion Lineal Simple m2 cubiertos	0.5476	0.5467
Regresion Lineal Simple Ambientes	0.3892	0.4573
Regresion Lineal Simple M2	0.3397	0.3051
Regresion Lineal Simple Cochera	0.0191	0.0207
Regresion Lineal Simple Estrenar	0.0044	0.0049



Como vemos en el gráfico anterior, los modelos de Regresión Lineal que mejor se ajustan a nuestro Dataset son los de Ridge y Lasso, con un R2 de aproximadamente 0.71.

CONCLUSIONES FINALES:

- **Los modelos que mejor se ajustan a nuestro Dataset son los de Ridge y Lasso, por tener los mayores R2.**

- **Las Regresiones Lineales Simples son una aproximación muy ineficiente, y a medida que fuimos complejizando los modelos utilizando Regresiones Múltiples, y luego modelos de Regularización, logramos mejorar considerablemente los R2, y por ende mejorando la capacidad predictiva de los modelos.**

- **Como vemos, todos los R2 de TEST son mayores que los R2 de TRAIN, por lo que los modelos no overfittean.**

- **Utilizar el algoritmo de DBSCAN para detectar Outliers (labels=-1), utilizando las variables "surface_total_in_m2" y "Ambientes, nos permitió mejorar mucho el R2 y por ende la precisión de los modelos.**

    - **Imputar valores en los registros que tenían nulls en la columna "Ambientes" nos ayudó también a mejorar el R2 y la precisión de los modelos.**