Proyecto Final de Inteligencia Artificial
“Predicción de Precios de bienes”

Alumno
Libardo Andres Carpio Sepúlveda

Ingeniero/Profesor
Carlos Francisco Calderón PhD.

Pontificia Universidad Javeriana
Ingeniería Electrónica
Bogotá D.C.
2023

Introducción

Hoy en día podemos apreciar como el mercado es muy volátil por diversos temas, políticos, ambientales y sociales, todas estas cosas afectan nuestros bolsillos, pues nuestros bienes y cambian su valor con respecto a eso.
Esto supone un gran problema para todos nosotros, pues cuando nos hemos esforzado por comprar cosas el mercado va cambiando su valor, un breve ejemplo y muy común es una casa, un apartamento o un carro.

Hoy nos vamos a centrar en un carro, el cual es un bien muy común y que día a día se comercializan por montón, pero llegamos al dilema de cuánto cuesta en este momento nuestro vehículo y que tan rápido queremos que se realice nuestra venta, pues es de saber que a medida que nuestro cobramos más por nuestro bien, tendremos que esperar más para realizar nuestra venta y mucho más cuando hay tanta oferta.

Para ello, queremos brindar la ayuda a nuestros usuarios diseñando un cotizador, que nos ayuda a predecir y ajustar el precio de nuestro auto teniendo en cuenta ciertas características, y ajustándolo a un histórico de precios y su comportamiento a lo largo del tiempo, para que nuestros usuarios puedan vender su auto al mejor precio en relación con su tiempo de venta.

Con este proyecto tenemos como fin diseñar un algoritmo de aprendizaje automático que nos permita facilitar la vida de las personas

Algoritmo

Para el desarrollo de este proyecto estaremos manejando un algoritmo Supervisado, pues tendremos una cantidad “x” de características y una cantidad “y” de etiquetas.

Donde nuestras “x” serán 17 características que incluyen, marca, color, modelo, año de venta, año de registro, entre otros. Entre ellas tendremos variables numéricas, categóricas, fechas, y texto. Por lo que para poder usar nuestro algoritmo tendremos que realizar limpieza de características innecesarias y hacer un pre-procesado de las mismas para facilitar su tratamiento.

BBDD

Para este proyecto se hizo uso de una base de datos de Kaggle, la cual parecía muy completa y muy bien realizada lo cual nos ayudaba y facilitaba aun el trabajo de los datos.

	maker_key	model_key	mileage	engine_pow	registration_	fuel	paint_color	car_type	feature_1	feature_2	feature_3	feature_4	feature_5	feature_6	feature_7	feature_8	price	sold_at
2	BMW	118	140411	100	1/02/12	diesel	black	convertible	TRUE	TRUE	FALSE	FALSE	TRUE	TRUE	TRUE	FALSE	11300	1/01/18
3	BMW	M4	13929	317	1/04/16	petrol	grey	convertible	TRUE	TRUE	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	69700	1/02/18
4	BMW	320	183297	120	1/04/12	diesel	white	convertible	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	TRUE	FALSE	10200	1/02/18
5	BMW	420	128035	135	1/07/14	diesel	red	convertible	TRUE	TRUE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	25100	1/02/18
6	BMW	425	97097	160	1/12/14	diesel	silver	convertible	TRUE	TRUE	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	33400	1/04/18
7	BMW	335	152352	225	1/05/11	petrol	black	convertible	TRUE	TRUE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	17100	1/02/18
8	BMW	325	205219	145	1/05/09	diesel	grey	convertible	TRUE	TRUE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	12400	1/02/18
9	BMW	118	115560	105	1/08/09	petrol	white	convertible	TRUE	TRUE	FALSE	FALSE	FALSE	TRUE	TRUE	FALSE	6100	1/02/18
10	BMW	Z4	123886	125	1/07/04	petrol	black	convertible	TRUE	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE	FALSE	6200	1/03/18
11	BMW	320	139541	135	1/06/13	diesel	white	convertible	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	TRUE	FALSE	17300	1/03/18
12	BMW	320	77115	135	1/09/12	diesel	blue	convertible	TRUE	TRUE	FALSE	FALSE	TRUE	FALSE	TRUE	TRUE	19300	1/03/18
13	BMW	325	228000	145	1/09/09	diesel	black	convertible	TRUE	TRUE	FALSE	FALSE	TRUE	FALSE	TRUE	TRUE	13300	1/03/18
14	BMW	420	132025	135	1/03/14	diesel	blue	convertible	TRUE	TRUE	FALSE	FALSE	TRUE	TRUE	TRUE	FALSE	21700	1/03/18
15	BMW	420	77061	135	1/08/15	diesel	black	convertible	TRUE	TRUE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	36300	1/03/18
16	BMW	120	174631	120	1/01/09	diesel	black	convertible	TRUE	TRUE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	10500	1/04/18
17	BMW	120	208945	130	1/04/08	diesel	grey	convertible	TRUE	TRUE	FALSE	FALSE	TRUE	FALSE	TRUE	TRUE	8300	1/04/18
18	BMW	220	21167	135	1/07/15	petrol	white	convertible	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	TRUE	21900	1/04/18
19	BMW	650	24521	270	1/01/08	petrol	grey	convertible	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	9200	1/09/18
20	BMW	325	205474	145	1/05/09	diesel	grey	convertible	TRUE	TRUE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	11000	1/04/18
21	BMW	Z4	128940	110	1/03/08	petrol	black	convertible	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	7700	1/05/18
22	BMW	135	126213	225	1/02/13	petrol	white	convertible	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	16800	1/05/18
23	BMW	218	24868	100	1/01/16	petrol	grey	convertible	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE	31000	1/05/18
24	BMW	420	90401	135	1/10/14	diesel	grey	convertible	TRUE	TRUE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	26300	1/05/18
25	BMW	220	46963	140	1/10/15	diesel	orange	convertible	FALSE	TRUE	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	23300	1/05/18
26	BMW	Z4	43662	180	1/05/13	petrol	orange	convertible	TRUE	FALSE	FALSE	FALSE	TRUE	FALSE	TRUE	FALSE	20800	1/05/18
27	BMW	318	196092	85	1/10/95	petrol	blue	convertible	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	1800	1/05/18
28	BMW	430	113678	190	1/07/14	diesel	black	convertible	TRUE	TRUE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	30100	1/05/18
29	BMW	420	65988	135	1/07/14	diesel	grey	convertible	TRUE	TRUE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	28700	1/05/18
30	BMW	Z4	123809	125	1/07/04	petrol	black	convertible	TRUE	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE	FALSE	6500	1/05/18
31	BMW	320	181252	120	1/08/08	diesel	black	convertible	TRUE	TRUE	FALSE	FALSE	TRUE	FALSE	TRUE	TRUE	11100	1/05/18
32	BMW	118	195114	105	1/10/10	diesel	red	convertible	TRUE	TRUE	FALSE	FALSE	FALSE	TRUE	TRUE	FALSE	9900	1/05/18
33	BMW	Z4	82761	110	1/05/05	petrol	grey	convertible	TRUE	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE	FALSE	9500	1/05/18
34	BMW	218	25050	100	1/01/16	petrol	grey	convertible	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE	31000	1/06/18

bmw_pricing_challenge +

Imagen # 1. Base de datos para pre-procesar y analizar

Etapa de pre-procesado

En esta etapa es muy analítica por parte del ingeniero, pues se necesita que comenzar a usar la lógica, por ejemplo determinar que valores son erróneos, cuales son nulos, cuales no son coherentes, por ejeplo: Un motor no puede ser negativo, no puede tener -500cc, no existen kilómetros negativos, además como estamos hablando de bienes usados y en este caso carros, su kilometraje debe ser un numero positivo mayor a cero.

Toda esta estapa la vamos realizando con filtros de condiciones y eliminando aquellas filas de experimentos o muestras que no cumplen nuestras condiciones básicas.

```
# Se crea una condición booleana en la variable 'Clean_Motor' que
evalúa si el valor
# de la columna 'engine_power' en el DataFrame 'BBDD' es mayor que 1
(Positiva y
# teniendo en cuenta que el valor mínimo de un motor de automovil es
1000c.c.).

Clean_Motor = BBDD['engine_power'] > 1
datos = BBDD [Clean_Motor]

# La variable 'Clean_Motor' contendrá una serie booleana con True en
las posiciones
# donde se cumple la condición y False donde no.\
```

En el caso anterior podemos apreciar que estamos realizamos el recorte de esos datos que hemos mencionado en uuna de las variables, en este caso ‘Engine_Power’.

Ahora procedemos a graficar las distribuciones de nuestros datos, para que podamos conocer los outlaiers de nuestras características. Los outlaiers, son aquellos valores atípicos, por ejemplo en este caso estamos hablando de la venta de autos usados y una base de datos de casi 5000, por lo cual muchos de ellos serán autos muy comerciales, y van a estar concentrados en un rango de precios, mientras que si en esta BBDD tenemos autos gama alta y de lujo, muy seguramente será un valor atípico y lo consideramos un outliers, pues no se encuentra dentro de nuestra varianza y deberá ser eliminado de nuestra muestra de datos a tener en cuenta para realizar nuestra regresión.

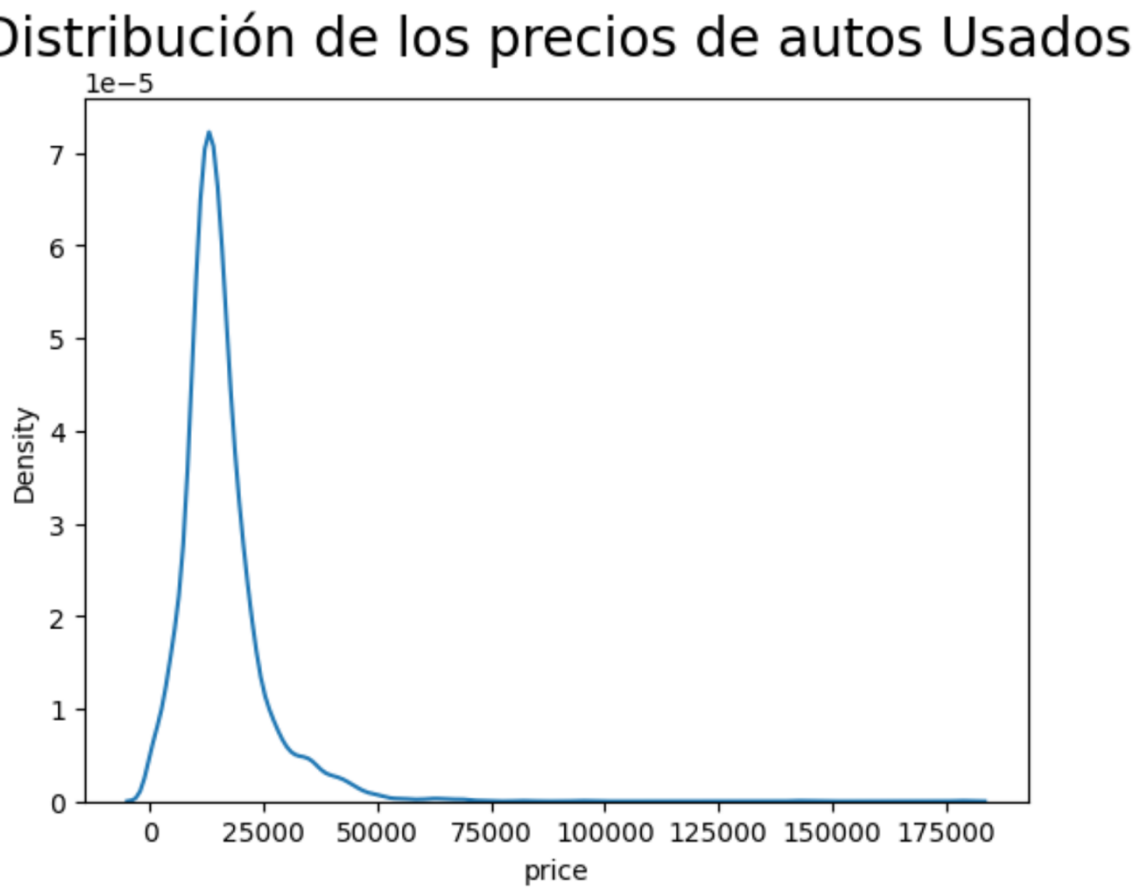

Imagen 2. Distribución de precios de autos usados

Como podemos observar, tenemos la mayor concentración de precios hasta los más o menos 60 mil a 70 mil, sin embargo, podemos ver que aún siguen existiendo datos hasta un poco más de los 175 mil, pero como lo mencionamos antes, son valores muy atípicos y lo podemos confirmar observando la densidad que tiene nuestra grafica para esos valores.

Por lo que procedemos a eliminar esos datos que se salen de nuestra concentración de valores.

Para ello lo que usaremos es valores de techo y piso, fijados manualmente por nosotros, lo cual realizamos de la siguiente forma.

```
#*****************************************************************
******************#
# Observando la grafica anterior, podemos decir que la mayor cantidad
de precios se
# concentran en valores hasta más o menos los 60000 USD, por lo que
podemos considerar
# que los demás precios son valores atípicos y podemos descartarlos.

# Nuestra cariable Precio_acotado contendrá una serie booleana con
True en las posiciones donde el precio sea mayor a
# 60000 y False donde no lo sea.
Precio_acotado = BBDD['price'] > 60000
# Contamos, cuántos registros cumplen con la condición de tener un
precio menor a 60000.


# Distribucion de kilometraje
sns.kdeplot(data=datos, x='mileage')
plt.title('Distribución de Kilometraje en autos Usados', size=20)
#*****************************************************************
******************#
```

Este estilo de limpieza se uso es todo el código, pues a lo largo del desarrollo del proyecto iremos realizando limpiezas y acondicionamiento de datos/variables.

Análisis de características

A continuación, podremos observar las gráficas con las cuales hemos podido guiarnos para realizar la limpieza de nuestros datos y eliminar outlaiers de nuestras características.

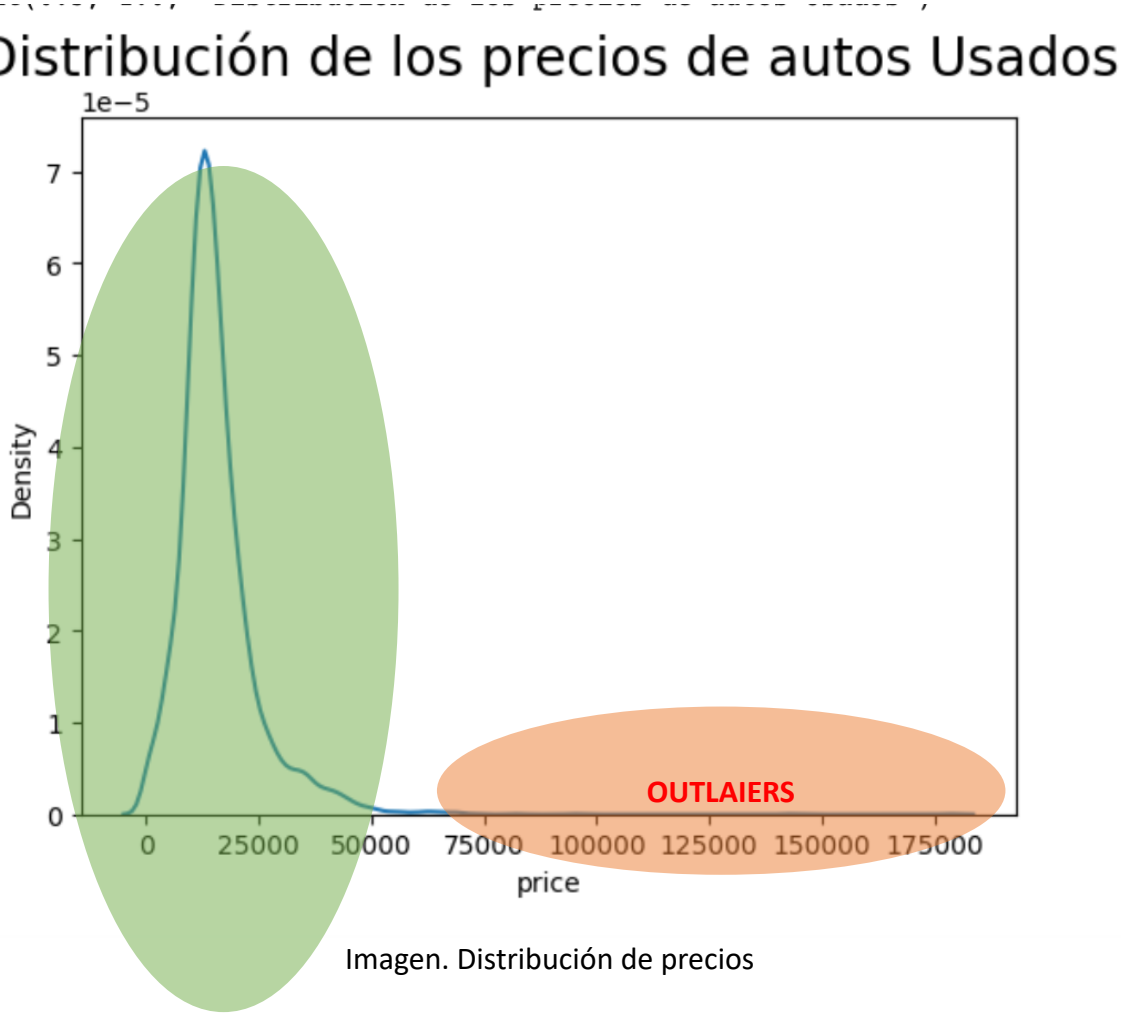

Imagen. Distribución de precios

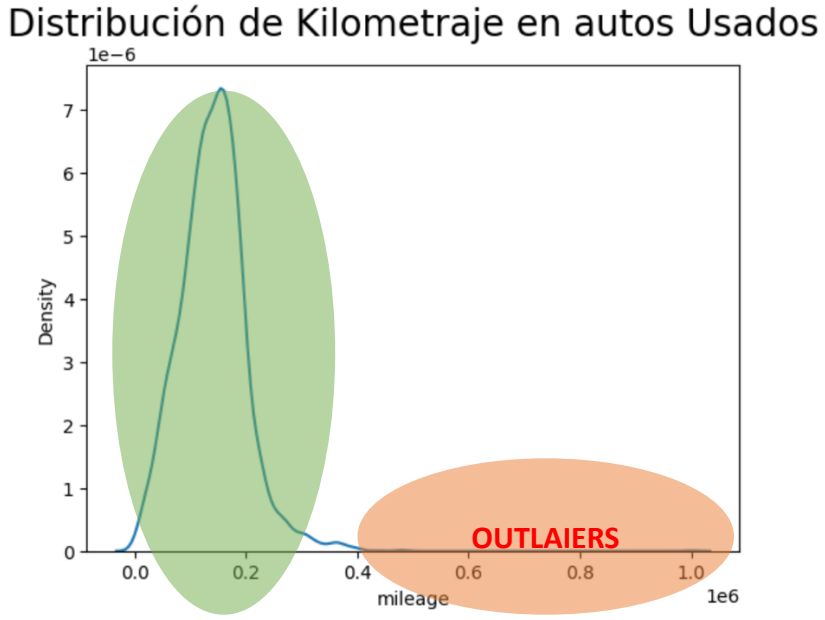

Imagen. Como se distribuyen los kilómetros en la muestra de autos

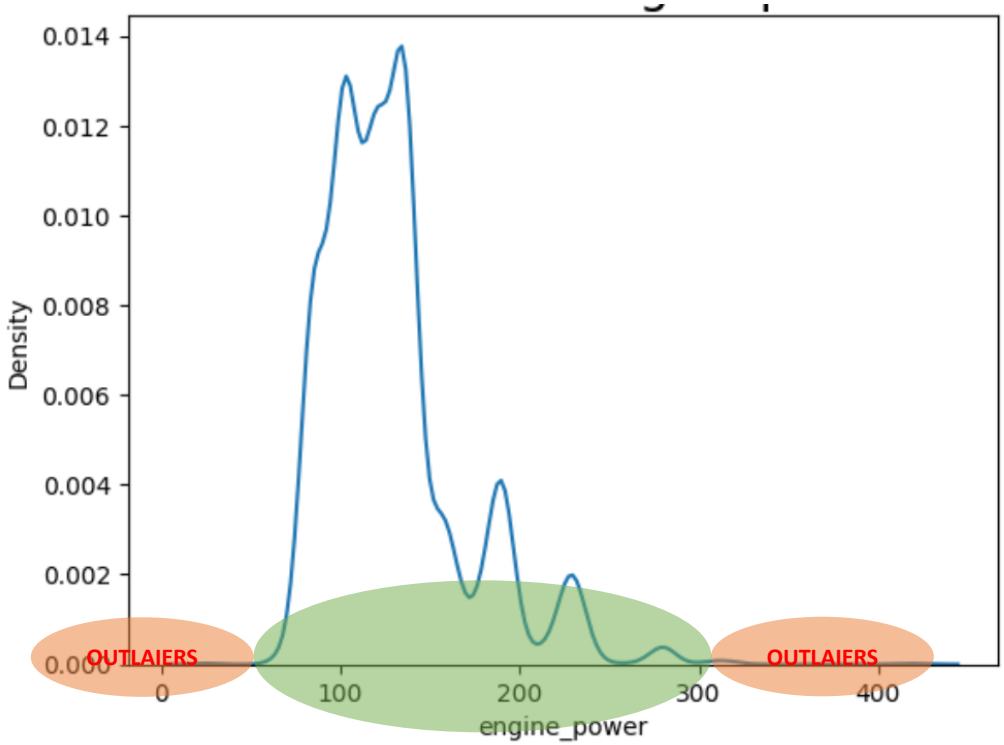

Imagen. Distribución de motorizaciones en la muestra de autos

Una vez hemos acondicionado y limpiado toda nuestra BBDD, procedemos a realizar graficas que nos permitan analizar nuestras características, y encontrar correlaciones entre ellas. Para este desarrollo usaremos los mapas de calor, pues nos brindan muy buena información de la cual podemos dar conclusiones a medida que avanzamos. Por ejemplo:

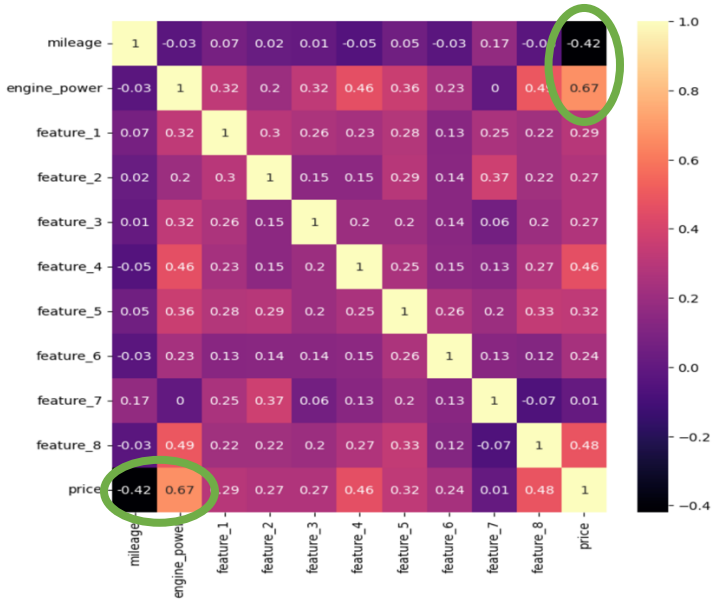

Imagen 3. Mapa de calor de nuestras características

Analizando la imagen nos podemos centrar de momento en dos valores muy importantes, como lo son la relación “Precio – Kilometraje” y “Precio – Engine Motor” pues estas características vemos que son inversamente proporcional y directamente proporcional respectivamente, donde podemos decir que a medida que el kilometraje de nuestro auto aumenta, el precio de nuestro auto disminuye; pero por otro lado cuando el motor de nuestro auto es más grande, mayor es el precio al que lo podemos vender.

Adicional a esta, tenemos otra característica que sabemos que son muy importantes al momento de comprar autos, y es el tiempo de uso, por lo que incluiremos esta característica en nuestro mapa de calor.

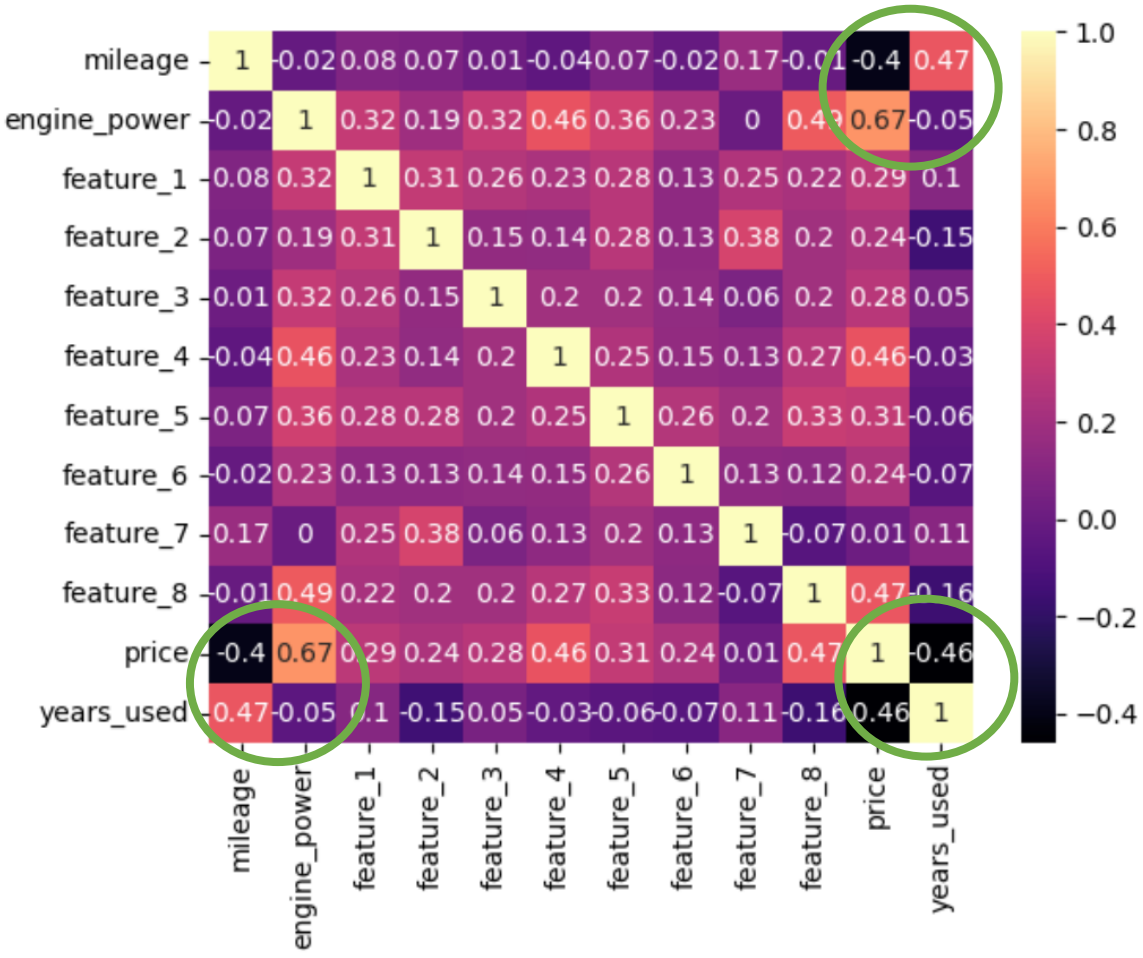

Imagen. Adición de característica a nuestro mapa de calor

Finalmente, cuando ya se realizó todo esta limpieza y pre-procesado de nuestros datos, podemos comenzar a realizar nuestro modelo de regresión, para el cual usaremos dos modelos, el modelo de regresión por sklearn y el modelo de random forest como lo podemos ver a continuación.

```
# creamos una instancia del modelo de regresión lineal.
LReg = LinearRegression()
# Se ajusta el modelo utilizando los conjuntos
LegReg = LReg.fit(X_train, Y_train)
# Se utiliza el modelo ajustado para hacer predicciones sobre los
datos de prueba
# en X_test. Se obtiene un vector de valores predichos para la
variable dependiente.
Y_pred=LReg.predict(X_test)
# Se utiliza la función r2_score() de scikit-learn para calcular el
coeficiente de
# determinación R^2, que indica la calidad de ajuste del modelo. Se
compara los valores
# reales Y_test con los valores predichos Y_pred para evaluar la
capacidad de predicción
# del modelo.
R2=r2_score(Y_test , Y_pred)
print ('R2:',R2.round(5))

# Lo mismo pero con escalado
X_test_scaled2 = StandardScaler().fit_transform(X_test)
X_train_scaled2 = StandardScaler().fit_transform(X_train)
LReg2=LinearRegression()
LReg2.fit(X_train_scaled2,Y_train)
Y_pred2=LReg2.predict(X_test_scaled2)
R22 = r2_score(Y_test , Y_pred2)
print('R2:',R22.round(5))

# Al escalar las características antes de ajustar el modelo de
regresión lineal, se
# busca asegurar una mejor interpretación de los coeficientes, mejorar
el rendimiento
# del modelo y reducir la influencia de valores atípicos.
```

Como podemos observar, calculamos los coeficientes de determinación para evaluar que tan buena es nuestra regresión, y los hallamos de dos formas, sin escalado y con escalado para poder ver la diferencia, posterior a ello procedemos a graficarla.

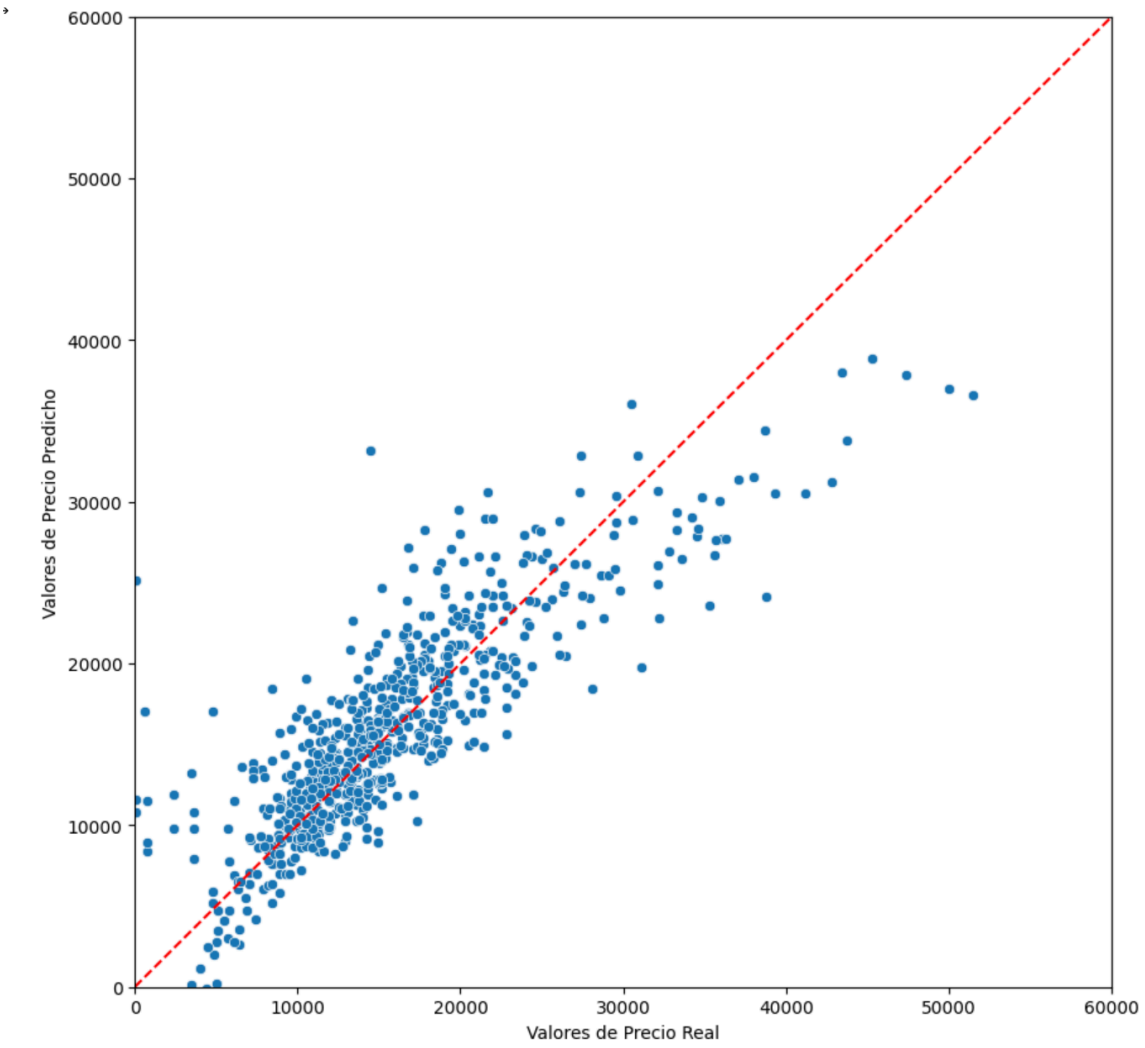

Imagen. Regresión Lineal Sklearn.

Para este modelo tenemos un. `R2: 0.72126 R2 Escalado: 0.71981`

Ahora procedemos a graficar la de Random Forest para ver cuál de las dos es mejor y a su vez, observar cual de los dos R2 es mejor, y cual modelo se ajusta más.

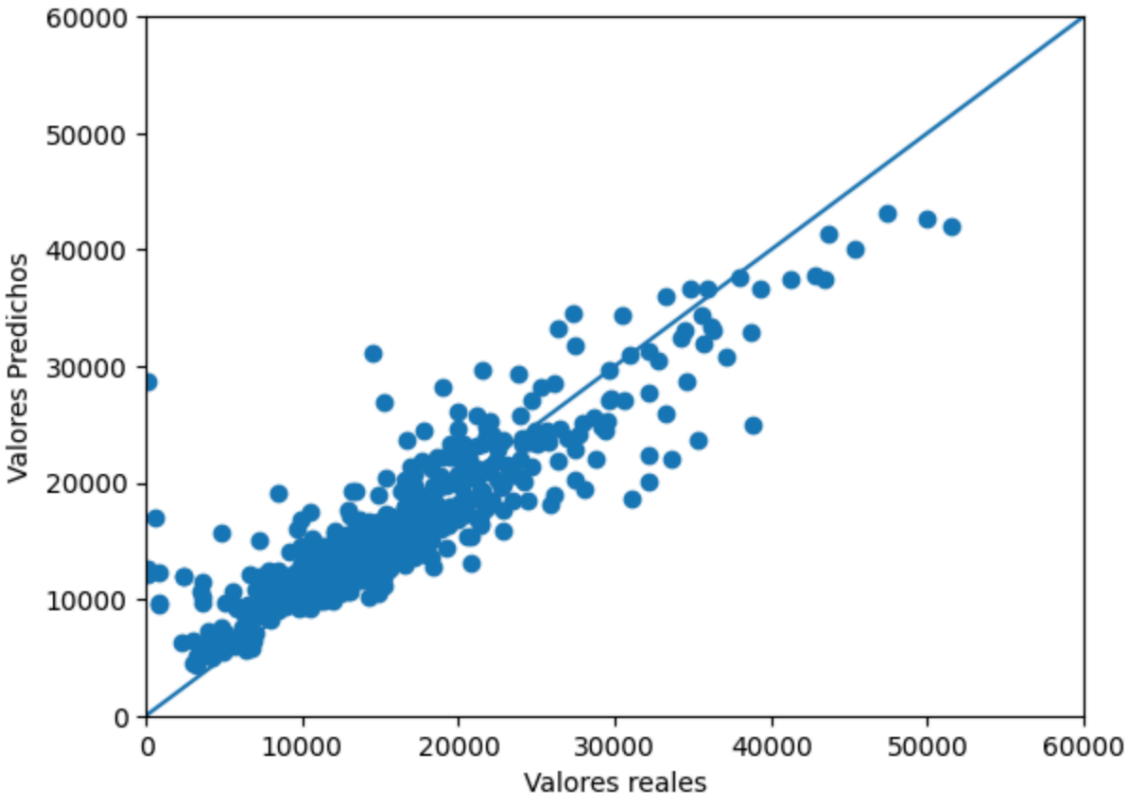

Imagen. Grafica con modelo Random Forest

Como podemos ver, y comprobar con el R2, este modelo se adapta mucho mejor a nuestros precios pues tenemos un `R2: 0.8`

Por lo que podemos decir que es un buen modelo de regresión, dado esto podemos decir que hemos logrado nuestro objetivo de realizar el diseño de un algoritmo que nos ayude a predecir el comportamiento del precio de nuestros vehículos a medida que pasa el tiempo y con sus características.

Como un plus, realizaremos más adelante la adaptación de una interfaz la cual nos ayude a que nuestros usuarios ingresen sus características y un algoritmo de aprendizaje automático pueda determinar las características que nos está entregando y nos devuelva el precio estimado de nuestros vehículos.

Adicional a esto el código ha sido pensado para aplicar a más bienes y se puede adaptar fácilmente a otras bases de datos sin ningún inconveniente.