# ENTREGABLE N°1

VIGEE CAROLINA ROJAS B.

## BMW PRICING
### PREPROCESSING

En este entregable, trataré de limpiar y analizar los datos aportados en el dataset, para lograr averiguar cuáles son las variables que influyen en el precio y así conseguir predecirlo en función de las mas relevantes.

### 1.ESTRUCTURA DE DATOS:

A través de estos códigos podemos observar que el dataset se compone de 4843 datos y 18 columnas de las cuales 1 es de bool, 3 son de floats y 14 de object, también se puede observar que no existen duplicados.

```
df_bmw.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4843 entries, 0 to 4842
Data columns (total 18 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   marca                       4841 non-null   object
 1   modelo                      4840 non-null   object
 2   km                          4841 non-null   float64
 3   potencia                    4842 non-null   float64
 4   fecha_registro              4842 non-null   object
 5   tipo_gasolina               4838 non-null   object
 6   color                       4831 non-null   object
 7   tipo_coche                  4834 non-null   object
 8   volante_regulable           4839 non-null   object
 9   aire_acondicionado          4841 non-null   object
 10  camara_trasera              4841 non-null   object
 11  asientos_traseros_plegables 4839 non-null   object
 12  elevalunas_electrico        4841 non-null   object
 13  bluetooth                   4839 non-null   object
 14  gps                         4843 non-null   bool
 15  alerta_lim_velocidad        4841 non-null   object
 16  precio                      4837 non-null   float64
 17  fecha_venta                 4842 non-null   object
dtypes: bool(1), float64(3), object(14)
memory usage: 648.1+ KB

[ ] df_bmw.shape

(4843, 18)
```

```
df_bmw.describe()
```

| | km | potencia | precio |
|---|---|---|---|
| count | 4.841000e+03 | 4842.000000 | 4837.000000 |
| mean | 1.409593e+05 | 128.981826 | 15831.920612 |
| std | 6.020853e+04 | 38.994839 | 9222.630708 |
| min | -6.400000e+01 | 0.000000 | 100.000000 |
| 25% | 1.028840e+05 | 100.000000 | 10800.000000 |
| 50% | 1.410800e+05 | 120.000000 | 14200.000000 |
| 75% | 1.752170e+05 | 135.000000 | 18600.000000 |
| max | 1.000376e+06 | 423.000000 | 178500.000000 |

### 2.DETECCIÓN DE NULOS:

En este dataset, sólo una variable (gps) no presenta nulos.

```
df_bmw2.isnull().sum()

marca                        2
modelo                       3
km                           2
potencia                     1
fecha_registro               1
tipo_gasolina                5
color                       12
tipo_coche                   9
volante_regulable            4
aire_acondicionado           2
camara_trasera               2
asientos_traseros_plegables  4
elevalunas_electrico         2
bluetooth                    4
gps                          0
alerta_lim_velocidad         2
precio                       6
fecha_venta                  1
dtype: int64
```

La primera columna es marca, como todos los coches son de la marca BMW, tomé la decisión de eliminarla ya que no proporciona ninguna información relevante que influya en la predicción del precio.
La segunda columna es modelo, tienen 3 nulos, los cuales decidí renombrar como “sin modelo”.
La tercera columna es km, en esta columna eliminaremos los 2 nulos, además como pudimos comprobar en el .describe habían km que estaban en negativo por lo que al observar el precio este debe ser un coche 0 km, es por ello que remplazaré este negativo con 0 usando la función np.where.

df_bmw2["km"]= np.where(df_bmw2["km"]< 0, 0, df_bmw2["km"])

La columna 4 corresponde a potencia, en esta variable existían coches con potencia menor a 70, para solucionarlo calculé la media entre 70 y 420 para así remplazar los valores menores a 70 por esta media, y el nulo fue remplazado también por la media.

```
#no debe haber coches con potencia menor a 70 por lo que sacamos la media

[ ] media_potencia =df_bmw2[(df_bmw2["potencia"] > 70) & (df_bmw2["potencia"]< 420)]["potencia"].mean()

[ ] df_bmw2["potencia"]= np.where(df_bmw2["potencia"]< 70, media_potencia, df_bmw2["potencia"])

[ ] df_bmw2["potencia"].fillna(df_bmw2["potencia"].mean(), inplace=True)
```

La columna 5 indica la fecha de registro del coche cuando el propietario lo compró, en esta variable había un nulo que eliminé.

Las columnas 6, 7 y 9 que proporcionan la información sobre el tipo de gasolina o combustible utilizado en los coches, el color y el tipo de coche , los nulos fueron remplazados por “sin tipo gasolina”, “sin color”, “sin tipo coche” respectivamente.

Las columnas 9 a la 16 brindan información sobre si los coches tienen o no ciertas características que podrían ser catalogadas como accesorios, en este caso los nulos fueron remplazados por la función mode.
df_bmw2["volante_regulable"].fillna(df_bmw2["volante_regulable"].mode()[0], inplace= True)

La columna 17 es el precio de venta del coche en la cual existían valores inferiores a 400, en este caso remplazamos estos precios inferiores por la media al igual que a los nulos.
La columna 18 es la fecha final en la que se vendió el coche supongo que al segundo propietario.

## 3.ANÁLISIS UNIVARIABLES.

En “tipo_gasolina” hay seis valores distintos: diesel, petrol, hybrid_petrol, sin tipo gasolina, Diesel y electro, como se puede observar hay dos valores iguales en la que sólo cambia la letra mayúscula, así que decidí renombrar la que tenía menor cantidad de coches quedando sólo el valor diesel.

Del mismo modo decidí centrarme solo en los coches de diesel y gasolina ya que la cantidad de coches eléctricos e híbridos es mucho menor, por lo que los traté como valores atípicos eliminándolos de los datos de la columna.

```
[ ] def replace_name(a,b):
        df_bmw3["tipo_gasolina"].replace(a,b,inplace=True)

    replace_name('Diesel','diesel')

df_bmw3["tipo_gasolina"].value_counts()

diesel              4626
petrol               191
hybrid_petrol          8
sin tipo gasolina      5
electro                3
Name: tipo_gasolina, dtype: int64

[ ] df_bmw3 = df_bmw3.drop(df_bmw3[(df_bmw3['tipo_gasolina'] == 'electro')| (df_bmw3['tipo_gasolina'] == 'hybrid_petrol') | (df_bmw3['tipo_gasolina'] == 'sin tipo gasolina')].index)

[ ] df_bmw3["tipo_gasolina"].value_counts()

diesel    4626
petrol     191
Name: tipo_gasolina, dtype: int64
```

En modelo, creé una fila llamada otros para agrupar los modelos con cantidad menor a 20, con ello conseguí pasar de tener 76 modelos a tener 24 .

```
[ ] #crear en modelos una fila que agrupe todos los modelos con cantidad menor a 20

[ ] Grupo_modelo = df_bmw3['modelo'].value_counts().to_frame()
    Grupo_modelo =  Grupo_modelo[Grupo_modelo['modelo'] < 20].index
    Grupo_modelo
    for i in Grupo_modelo:
      df_bmw3['modelo'] =df_bmw3.modelo.replace(
              to_replace =i,
              value = "otros")

[ ] len(df_bmw3['modelo'].unique())

    24
```

Las dos variables de fechas las convertí a date, para luego crear una columna llamada edad_coche que se generó como resultado de restar las columnas fecha_venta y fecha_registro.

En edad_coche resultaron algunos valores negativos que decidí eliminar, al igual que se han eliminado las columnas de fechas, porque ya extraje la edad de los coches y estas ya no me brindaran gran información

```
for i in ["fecha_registro","fecha_venta"]:
    df_bmw3[i]= pd.to_datetime(df_bmw3[i])
```

```
[ ] #resto el año de fecha_venta con fecha registro para obtener edad del coche
                    (variable) df_bmw3: Any | DataFrame | None
df_bmw3["edad_coche"]=((df_bmw3["fecha_venta"])- (df_bmw3["fecha_registro"]))/np.timedelta64(1,'Y')

[ ] df_bmw3["edad_coche"]=np.round(df_bmw3["edad_coche"]).astype(int)

[ ] df_bmw3[df_bmw3["edad_coche"]<0] #ver de eliminar outliers de edad coche o hacer una media y reemplazarlo

    Seleccionar celda          ⌘/Ctrl+Shift+S
    Enlace a celda
```

```
df_bmw3 = df_bmw3.drop(df_bmw3[df_bmw3["edad_coche"]<0].index)
```

```
for i in ("fecha_registro", "fecha_venta"):
  del(df_bmw3[i])
```

En la columna color realicé el mismo procedimiento que en modelo creando otro grupo llamado otros donde están los colores que tengan una cantidad menor a 50.

```
[ ] len(df_bmw3['color'].unique())

    11

    Grupo_color = df_bmw3['color'].value_counts().to_frame()
    Grupo_color =  Grupo_color[Grupo_color['color'] < 50].index
    Grupo_color
    for i in Grupo_color:
      df_bmw3['color'] =df_bmw3.color.replace(
              to_replace =i,
              value = "otros")

[ ] len(df_bmw3['color'].unique())

    8
```

# ENTREGABLE N°1

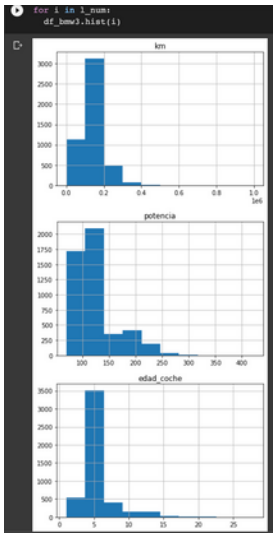
## 4 .SEPARAR VARIABLES EN : TARGET, CATEGÓRICAS, BOOLEAN Y NUMÉRICAS.

```
target = ["precio"]

def obtener_lista_variables(dataset):
    lista_numericas=[]
    lista_boolean=[]
    lista_categoricas=[]
    for i in dataset:
      if (dataset[i].dtype.kind == "i" or dataset[i].dtype.kind == "f") and (i not in target) and (len(dataset[i].unique()) !=2):
        lista_numericas.append(i)
      elif (dataset[i].dtype.kind == "b" or dataset[i].dtype.kind == "f") and (i not in target):
        lista_boolean.append(i)
      elif (dataset[i].dtype.kind == "O") and (i not in target):
          lista_categoricas.append(i)
    return lista_numericas, lista_boolean, lista_categoricas

l_num, l_bool, l_cat= obtener_lista_variables(df_bmw3)
```

```
l_num, l_bool, l_cat= obtener_lista_variables(df_bmw3)

l_num

['km', 'potencia', 'edad_coche']

l_bool

['volante_regulable',
 'aire_acondicionado',
 'camara_trasera',
 'asientos_traseros_plegables',
 'elevalunas_electrico',
 'bluetooth',
 'gps',
 'alerta_lim_velocidad']

l_cat

['modelo', 'tipo_gasolina', 'color', 'tipo_coche']
```

Para las variables numéricas realicé histogramas y tanto para las variables categóricas como las boolean un value_counts()

```
for i in l_num:
    df_bmw3.hist(i)
```




```
for i in l_bool:
  print(df_bmw3[i].value_counts())
```

```
True     2647
False    2166
Name: volante_regulable, dtype: int64
True     3816
False     997
Name: aire_acondicionado, dtype: int64
False    3843
True      970
Name: camara_trasera, dtype: int64
False    3854
True      959
Name: asientos_traseros_plegables, dtype: int64
False    2594
True     2219
Name: elevalunas_electrico, dtype: int64
False    3648
True     1165
Name: bluetooth, dtype: int64
True     4488
False     325
Name: gps, dtype: int64
True     2599
False    2214
Name: alerta_lim_velocidad, dtype: int64
```

```
for i in l_cat:
  print(df_bmw3[i].value_counts())
```

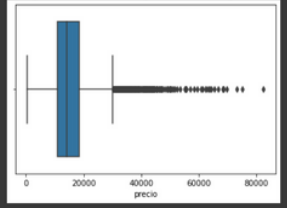
```
320                747
520                631
318                563
X3                 437
116                358
otros              288
X1                 274
316                233
X5                 229
525                183
530                157
118                143
318 Gran Turismo    97
320 Gran Turismo    73
518                 66
```

## 5.ANÁLISIS DE TARGET

En precio todavía existen  valores atípicos muy elevados por lo que remplazaré estos valores por la media calculada anteriormente.

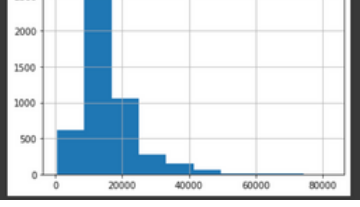
```
df_bmw3["precio"]= np.where(df_bmw3["precio"] > 85000, media_precio, df_bmw3["precio"])

sns.boxplot(x= df_bmw3["precio"])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff24e04c0d0>
```

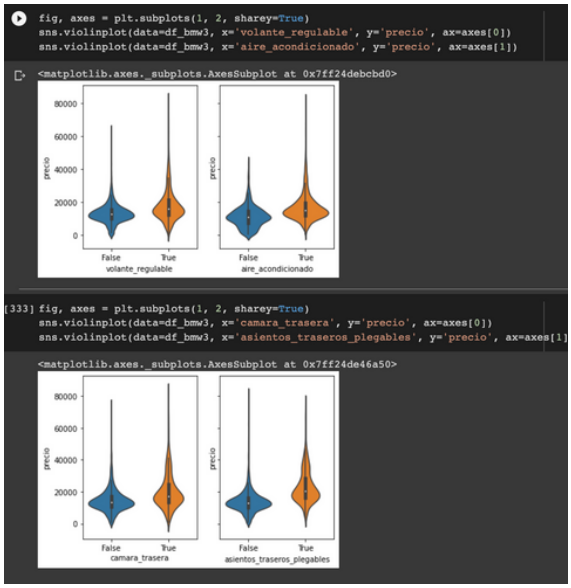



```
df_bmw3["precio"].hist()
```

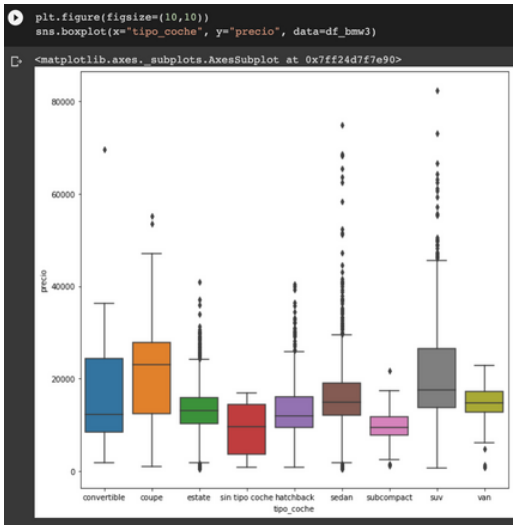
```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff24e038290>
```

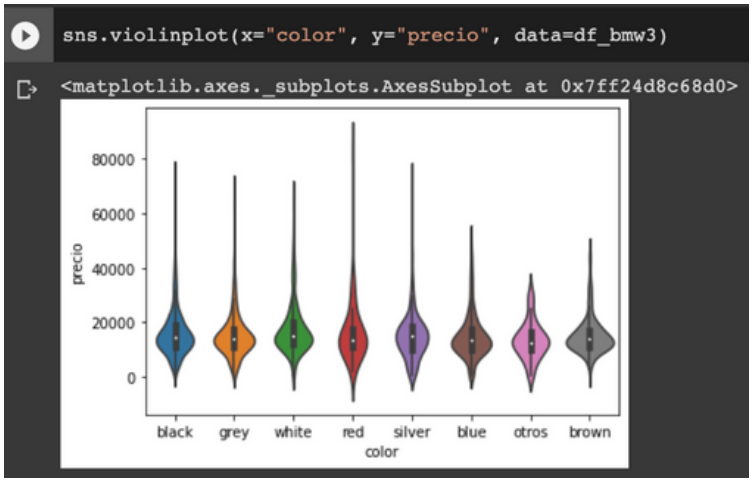


Al analizar la columna precio con las boolean podemos concluir que los coches que tienen estas características sl presentan una variación en el precio que podría ser relevante para predecir su precio.

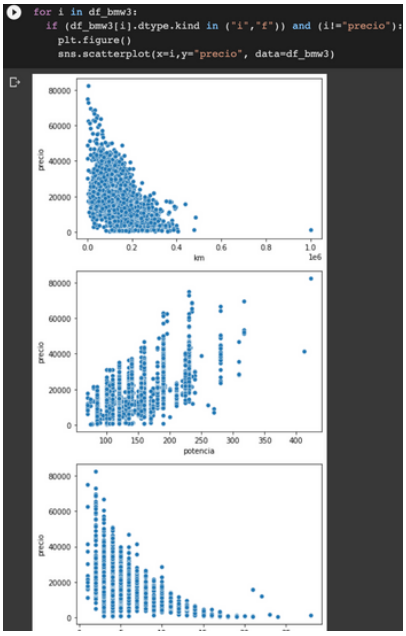



En las variables categóricas como modelo y tipo de coche también es posible observar una variación importante en el precio, sin embargo, en la columna color no sucede lo mismo, evidenciándose que dicha columna no es lo suficientemente relevante y útil al momento de predecir el precio.

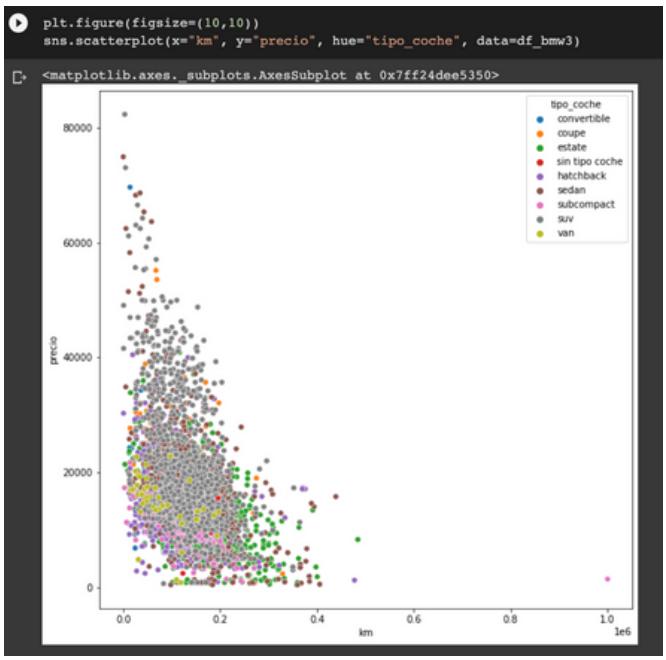



Las variables numéricas **potencia**, **km** y **edad_coche** por lo evidenciado en los diagramas tienen más relación con la variable precio. En potencia se puede ver que cuanto mayor es la potencia más elevado será el precio por lo que está positivamente correlacionada con el precio, caso contrario ocurre en las variables km y edad_coche ya que a menor kilometraje o edad el precio será más bajo, estando ambas negativamente correlacionadas con el precio.

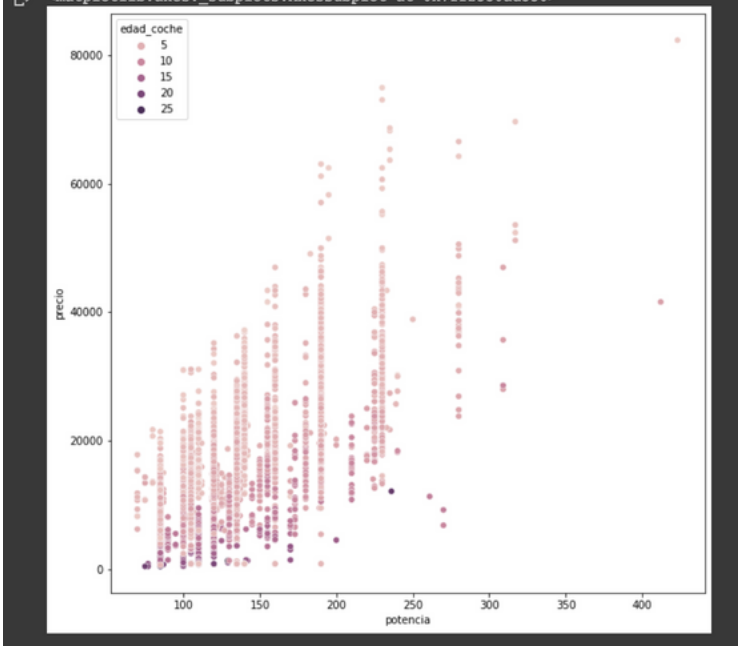



En el siguiente gráfico se demuestra que hay mayor cantidad de coches suv y que estos y los sedan presentan un precio medio más alto que otros tipos de coches. Vemos que la mayoría de los coches vendidos se oscilan entre los 10.000 y 35.000.



```
plt.figure(figsize=(10,10))
sns.scatterplot(x="potencia", y="precio", hue="edad_coche", data=df_bmw3)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff2530ad350>
```




Se puede ver que los coches con una edad inferior a 5 años son los que tienen un precio más elevado, además de que hay mayor cantidad de coches con una edad inferior a los 5 años , y los que tienen una potencia superior en la mayoría de los casos también presentan los precios más altos.

## 6.APLICAR FUNCIÓN GET_DUMMIES PARA LAS VARIABLES CATEGÓRICAS.

```
[350] df_bmw3 = pd.get_dummies(data=df_bmw3,columns=l_cat)
```

```
df_bmw3.head()
```

| | km | potencia | volante_regulable | aire_acondicionado | camara_trasera | asientos_traseros_plegables | elevalunas_electrico | bluetooth | gps | alerta_lim_velocidad | ... | color_white | tipo_coche_co |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 140411.0 | 100.0 | True | True | False | False | True | True | True | False | ... | 0 | |
| 1 | 13929.0 | 317.0 | True | True | False | False | False | True | True | True | ... | 0 | |
| 2 | 183297.0 | 120.0 | False | False | False | False | True | False | True | False | ... | 1 | |
| 3 | 128035.0 | 135.0 | True | True | False | False | True | True | True | True | ... | 0 | |
| 4 | 97097.0 | 160.0 | True | True | False | False | False | True | True | True | ... | 0 | |

5 rows × 55 columns

Se crean columnas por cada valor diferente de cada celda quedando valores numéricos.

En el caso de las variables boolean apliqué el siguiente código para dejar todas las variables en 0 y 1.

```
columnas_booleanas = ['volante_regulable','aire_acondicionado', 'camara_trasera', 'asientos_traseros_plegables', 'elevalunas_electrico','bluetooth','gps','alerta_lim_velocidad']
for columna in columnas_booleanas :
  df_bmw3[columna] =df_bmw3[columna].astype(int)
```

Se crea una correlación entre las variables con la variable precio.

```
corr.style.background_gradient(cmap='coolwarm')
```

| | km | potencia | volante_regulable | aire_acondicionado | camara_trasera | asientos_traseros_plegables | elevalunas_electrico | bluetooth | gps | alerta_lim_velocidad | precio | edad_coche |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| km | 1.000000 | -0.051067 | 0.068723 | 0.010687 | 0.002243 | -0.052877 | 0.045713 | -0.030472 | 0.154504 | -0.039903 | -0.405884 | 0.503221 |
| otencia | -0.051067 | 1.000000 | 0.326200 | 0.200692 | 0.316963 | 0.448022 | 0.338279 | 0.232583 | 0.004538 | 0.490749 | 0.645569 | -0.081972 |
| te_regulable | 0.068723 | 0.326200 | 1.000000 | 0.307425 | 0.253537 | 0.230627 | 0.277834 | 0.133859 | 0.244225 | 0.224261 | 0.265612 | 0.076389 |
| ondicionado | 0.010687 | 0.200692 | 0.307425 | 1.000000 | 0.150722 | 0.149727 | 0.284542 | 0.135646 | 0.371190 | 0.223606 | 0.246998 | -0.235482 |
| ara_trasera | 0.002243 | 0.316963 | 0.253537 | 0.150722 | 1.000000 | 0.200628 | 0.198236 | 0.144149 | 0.060890 | 0.201819 | 0.256191 | 0.022373 |
| aseros_plegables | -0.052877 | 0.448022 | 0.230627 | 0.149727 | 0.200628 | 1.000000 | 0.249247 | 0.154072 | 0.125944 | 0.277764 | 0.418492 | -0.057657 |
| nas_electrico | 0.045713 | 0.338279 | 0.277834 | 0.284542 | 0.198236 | 0.249247 | 1.000000 | 0.254843 | 0.202381 | 0.332636 | 0.262682 | -0.094842 |
| luetooth | -0.030472 | 0.232583 | 0.133859 | 0.135646 | 0.144149 | 0.154072 | 0.254843 | 1.000000 | 0.128875 | 0.120593 | 0.213405 | -0.074188 |
| gps | 0.154504 | 0.004538 | 0.244225 | 0.371190 | 0.060890 | 0.125944 | 0.202381 | 0.128875 | 1.000000 | -0.063963 | -0.003318 | 0.077454 |
| lim_velocidad | -0.039903 | 0.490749 | 0.224261 | 0.223606 | 0.201819 | 0.277764 | 0.332636 | 0.120593 | -0.063963 | 1.000000 | 0.446483 | -0.182140 |
| precio | -0.405884 | 0.645569 | 0.265612 | 0.246998 | 0.256191 | 0.418492 | 0.262682 | 0.213405 | -0.003318 | 0.446483 | 1.000000 | -0.439643 |
| ad_coche | 0.503221 | -0.081972 | 0.076389 | -0.235482 | 0.022373 | -0.057657 | -0.094842 | -0.074188 | 0.077454 | -0.182140 | -0.439643 | 1.000000 |

Como puede observarse, no parece haber una correlación entre sí de manera significativa, salvo las variables km y edad_coche que están relacionadas negativamente y la potencia que está correlacionada positivamente como ya lo había comentado anteriormente

Con anterioridad habíamos observado que el color del coche no aporta un factor determinante para predecir el precio, así que elimino las columnas creadas para color.

```
[360] for i in ("color_black", "color_blue","color_brown", "color_grey", "color_otros", "color_red", "color_silver", "color_white"):
        del(df_bmw3[i])
```

Por último aplico la función del minMaxScaler a las variables numéricas para asignarles un valor entre 0 y 1.

```
[362] minMaxResultado = MinMaxScaler()

[363] df_bmw3["km"]= minMaxResultado.fit_transform(df_bmw3["km"].values.reshape(-1,1))

[364] df_bmw3["potencia"]= minMaxResultado.fit_transform(df_bmw3["potencia"].values.reshape(-1,1))

[365] df_bmw3["edad_coche"]= minMaxResultado.fit_transform(df_bmw3["edad_coche"].values.reshape(-1,1))

df_bmw3.head()
```

| | km | potencia | volante_regulable | aire_acondicionado | camara_trasera | asientos_traseros_plegables | elevalunas_electrico | bluetooth | gps | alerta_lim_velocidad | ... | tipo_gasolina_petrol | tipo_coche_ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0.140358 | 0.084986 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | ... | 0 | |
| 1 | 0.013924 | 0.699717 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | ... | 1 | |
| 2 | 0.183228 | 0.141643 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | ... | 0 | |
| 3 | 0.127987 | 0.184136 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | ... | 0 | |
| 4 | 0.097061 | 0.254958 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | ... | 0 | |

5 rows × 47 columns