Import Librairies

In [1]:
```
import pandas as pd
```

Preparation Data

In [2]:
```
data = pd.read_csv("car_price_prediction.csv",sep=",")
data
```

Out[2]:

	ID	Price	Levy	Manufacturer	Model	Prod. year	Category	Leather interior
0	45654403	13328	1399	LEXUS	RX 450	2010	Jeep	Yes
1	44731507	16621	1018	CHEVROLET	Equinox	2011	Jeep	No
2	45774419	8467	-	HONDA	FIT	2006	Hatchback	No
3	45769185	3607	862	FORD	Escape	2011	Jeep	Yes
4	45809263	11726	446	HONDA	FIT	2014	Hatchback	Yes
...	...	...	...	...	...	...	...	...
19232	45798355	8467	-	MERCEDES-BENZ	CLK 200	1999	Coupe	Yes
19233	45778856	15681	831	HYUNDAI	Sonata	2011	Sedan	Yes
19234	45804997	26108	836	HYUNDAI	Tucson	2010	Jeep	Yes
19235	45793526	5331	1288	CHEVROLET	Captiva	2007	Jeep	Yes
19236	45813273	470	753	HYUNDAI	Sonata	2012	Sedan	Yes

19237 rows × 18 columns

In [3]:
```
data.sample(5)
```

Out[3]:

	ID	Price	Levy	Manufacturer	Model	Prod. year	Category	Leather interior
15927	45767429	9252	919	MERCEDES-BENZ	E 350	2012	Sedan	Yes
15845	45654191	282	1598	MERCEDES-BENZ	ML 350	2008	Jeep	Yes
2960	43973095	12000	2670	HONDA	Elysion	2010	Minivan	No
12504	45452333	14113	642	FORD	C-MAX	2012	Sedan	Yes
18387	45801945	14426	528	HYUNDAI	Elantra	2014	Sedan	Yes

In [4]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19237 entries, 0 to 19236
Data columns (total 18 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   ID                19237 non-null  int64  
 1   Price             19237 non-null  int64  
 2   Levy              19237 non-null  object 
 3   Manufacturer      19237 non-null  object 
 4   Model             19237 non-null  object 
 5   Prod. year        19237 non-null  int64  
 6   Category          19237 non-null  object 
 7   Leather interior  19237 non-null  object 
 8   Fuel type         19237 non-null  object 
 9   Engine volume     19237 non-null  object 
 10  Mileage           19237 non-null  object 
 11  Cylinders         19237 non-null  float64
 12  Gear box type     19237 non-null  object 
 13  Drive wheels      19237 non-null  object 
 14  Doors             19237 non-null  object 
 15  Wheel             19237 non-null  object 
 16  Color             19237 non-null  object 
 17  Airbags           19237 non-null  int64  
dtypes: float64(1), int64(4), object(13)
memory usage: 2.6+ MB
```

In [5]:
```
data.describe().round(2)
```

Out[5]:

	ID	Price	Prod. year	Cylinders	Airbags
count	19237.00	19237.00	19237.00	19237.00	19237.00
mean	45576535.89	18555.93	2010.91	4.58	6.58
std	936591.42	190581.27	5.67	1.20	4.32
min	20746880.00	1.00	1939.00	1.00	0.00
25%	45698374.00	5331.00	2009.00	4.00	4.00
50%	45772308.00	13172.00	2012.00	4.00	6.00
75%	45802036.00	22075.00	2015.00	4.00	12.00
max	45816654.00	26307500.00	2020.00	16.00	16.00

In [6]:
```
#duplication des lignes
data.duplicated().sum()
```

Out[6]: 313

In [7]:
```
#Affichage les lignes duppliqué
data[data.duplicated()]
```

Out[7]:

	ID	Price	Levy	Manufacturer	Model	Prod. year	Category	Leather interior
425	45815372	7840	-	MERCEDES-BENZ	E 200	1998	Sedan	No
1581	45815363	8781	-	TOYOTA	Ist	2002	Hatchback	No
1697	45732125	18503	-	TOYOTA	Prius	2010	Sedan	No
1972	45507765	20385	-	MERCEDES-BENZ	CLS 500	2005	Sedan	Yes
2131	45815363	8781	-	TOYOTA	Ist	2002	Hatchback	No
...	...	...	...	...	...	...	...	...
18974	45815465	2352	1273	LEXUS	IS 350	2015	Sedan	Yes
19137	45810466	15125	642	CHEVROLET	Orlando	2012	Jeep	Yes
19200	45725908	36065	640	MERCEDES-BENZ	CLA 250 AMG	2013	Sedan	Yes
19203	45761487	13485	645	TOYOTA	Prius	2011	Sedan	No
19210	45732939	8311	642	HYUNDAI	Sonata	2012	Sedan	Yes

313 rows × 18 columns

In [8]:
```
#Supprimer les lignes duppliqué
data = data.drop_duplicates()
```

In [9]:
```
data.duplicated().sum()
```

Out[9]: 0

In [10]:
```
#unique values
data.nunique()
```

Out[10]:
```
ID                 18924
Price               2315
Levy                 559
Manufacturer          65
Model               1590
Prod. year            54
Category              11
Leather interior       2
Fuel type              7
Engine volume        107
Mileage             7687
Cylinders             13
Gear box type          4
Drive wheels           3
Doors                  3
Wheel                  2
Color                 16
Airbags               17
dtype: int64
```

In [11]:
```
#Contenu de quelques colonnes
data[['Leather interior','Gear box type','Drive wheels','Doors','Wheel']]
```

Out[11]:
```
Leather interior                                    [Yes, No]
Gear box type       [Automatic, Tiptronic, Variator, Manual]
Drive wheels                                 [4x4, Front, Rear]
Doors                                     [04-May, 02-Mar, >5]
Wheel                          [Left wheel, Right-hand drive]
dtype: object
```

In [12]:
```
data['Levy'].info() #Levy doit etre un entier
```

```
<class 'pandas.core.series.Series'>
Int64Index: 18924 entries, 0 to 19236
Series name: Levy
Non-Null Count  Dtype
--------------  -----
18924 non-null  object
dtypes: object(1)
memory usage: 295.7+ KB
```

In [13]:
```
valeurs_non_entieres_rencontrees = set()

for index, valeur in data['Levy'].iteritems():
    if not str(valeur).isdigit() and valeur not in valeurs_non_entieres_r
        print("Valeur:", valeur)
        valeurs_non_entieres_rencontrees.add(valeur)
```

```
Valeur: -
/var/folders/kv/n_x4sgf13w90h65_x105kmv00000gn/T/ipykernel_25148/273307490
9.py:3: FutureWarning: iteritems is deprecated and will be removed in a fu
ture version. Use .items instead.
  for index, valeur in data['Levy'].iteritems():
```

In [14]:
```
valeurs_a_remplacer = {'-': -1} #-1 ca veut dire valeur manquantes
data['Levy'] = data['Levy'].replace(valeurs_a_remplacer)
```

```
/var/folders/kv/n_x4sgf13w90h65_x105kmv00000gn/T/ipykernel_25148/263585166
7.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  data['Levy'] = data['Levy'].replace(valeurs_a_remplacer)
```

In [15]:
```
#Convert to int
data['Levy'] = data['Levy'].astype(int)
```

```
/var/folders/kv/n_x4sgf13w90h65_x105kmv00000gn/T/ipykernel_25148/336366440
0.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  data['Levy'] = data['Levy'].astype(int)
```

In [16]:
```
data.sample(5)
```

Out[16]:

	ID	Price	Levy	Manufacturer	Model	Prod. year	Category	Leather interior
10228	45798191	11992	1047	HYUNDAI	Elantra	2007	Sedan	Yes
15076	45815192	8781	-1	MERCEDES-BENZ	C 230	2000	Sedan	No
5972	45507111	15950	-1	TOYOTA	Prius C	2013	Hatchback	No
4150	45772734	470	924	HONDA	Hr-v	2017	Jeep	Yes
8233	45784446	18189	738	CHEVROLET	Cruze	2017	Sedan	No

In [17]:
```
print("Min date prod year",data['Prod. year'].min(),"\t MAx date prod est
```

```
Min date prod year 1939          MAx date prod est  2020
```

In [18]:
```
data['Engine volume'].unique()
```

Out[18]:
```
array(['3.5', '3', '1.3', '2.5', '2', '1.8', '2.4', '4', '1.6', '3.3',
       '2.0 Turbo', '2.2 Turbo', '4.7', '1.5', '4.4', '3.0 Turbo',
       '1.4 Turbo', '3.6', '2.3', '1.5 Turbo', '1.6 Turbo', '2.2',
       '2.3 Turbo', '1.4', '5.5', '2.8 Turbo', '3.2', '3.8', '4.6', '1.
2',
       '5', '1.7', '2.9', '0.5', '1.8 Turbo', '2.4 Turbo', '3.5 Turbo',
       '1.9', '2.7', '4.8', '5.3', '0.4', '2.8', '3.2 Turbo', '1.1',
       '2.1', '0.7', '5.4', '1.3 Turbo', '3.7', '1', '2.5 Turbo', '2.6',
       '1.9 Turbo', '4.4 Turbo', '4.7 Turbo', '0.8', '0.2 Turbo', '5.7',
       '4.8 Turbo', '4.6 Turbo', '6.7', '6.2', '1.2 Turbo', '3.4',
       '1.7 Turbo', '6.3 Turbo', '2.7 Turbo', '4.3', '4.2', '2.9 Turbo',
       '0', '4.0 Turbo', '20', '3.6 Turbo', '0.3', '3.7 Turbo', '5.9',
       '5.5 Turbo', '0.2', '2.1 Turbo', '5.6', '6', '0.7 Turbo',
       '0.6 Turbo', '6.8', '4.5', '0.6', '7.3', '0.1', '1.0 Turbo', '6.
3',
       '4.5 Turbo', '0.8 Turbo', '4.2 Turbo', '3.1', '5.0 Turbo', '6.4',
       '3.9', '5.7 Turbo', '0.9', '0.4 Turbo', '5.4 Turbo', '0.3 Turbo',
       '5.2', '5.8', '1.1 Turbo'], dtype=object)
```

In [19]:
```
#Mileage convert to int et change name to Mileage(km)

#check if all columns contain km
(data['Mileage'].str.contains('km')).sum() == len(data['Mileage'])
```

Out[19]:
```
True
```

In [20]:
```
#Renommez la colonnes
data.rename(columns={'Mileage': 'Mileage (Km)'}, inplace=True)
data.columns.tolist()
```

```
/var/folders/kv/n_x4sgf13w90h65_x105kmv00000gn/T/ipykernel_25148/342581430
8.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  data.rename(columns={'Mileage': 'Mileage (Km)'}, inplace=True)
```

Out[20]:
```
['ID',
 'Price',
 'Levy',
 'Manufacturer',
 'Model',
 'Prod. year',
 'Category',
 'Leather interior',
 'Fuel type',
 'Engine volume',
 'Mileage (Km)',
 'Cylinders',
 'Gear box type',
 'Drive wheels',
 'Doors',
 'Wheel',
 'Color',
 'Airbags']
```

In [21]:
```
#retirer km
data['Mileage (Km)'] = data['Mileage (Km)'].str.replace(' km', '')
```

```
#convert to int
data['Mileage (Km)']=data['Mileage (Km)'].astype(int)
```

```
/var/folders/kv/n_x4sgf13w90h65_x105kmv00000gn/T/ipykernel_25148/20532592
1.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  data['Mileage (Km)'] = data['Mileage (Km)'].str.replace(' km', '')
/var/folders/kv/n_x4sgf13w90h65_x105kmv00000gn/T/ipykernel_25148/20532592
1.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  data['Mileage (Km)']=data['Mileage (Km)'].astype(int)
```

In [22]:
```
data['Cylinders'].info()
```

```
<class 'pandas.core.series.Series'>
Int64Index: 18924 entries, 0 to 19236
Series name: Cylinders
Non-Null Count  Dtype
--------------  -----
18924 non-null  float64
dtypes: float64(1)
memory usage: 295.7 KB
```

In [23]:
```
#colonnes cylindre
data['Cylinders'].unique()
```

Out[23]:
```
array([ 6.,  4.,  8.,  1., 12.,  3.,  2., 16.,  5.,  7.,  9., 10., 14.])
```

In [24]:
```
#convert to int
data['Cylinders']=data['Cylinders'].astype(int)
```

```
/var/folders/kv/n_x4sgf13w90h65_x105kmv00000gn/T/ipykernel_25148/240218509
7.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  data['Cylinders']=data['Cylinders'].astype(int)
```

In [25]:
```
data['Cylinders'].info()
```

```
<class 'pandas.core.series.Series'>
Int64Index: 18924 entries, 0 to 19236
Series name: Cylinders
Non-Null Count  Dtype
--------------  -----
18924 non-null  int64
dtypes: int64(1)
memory usage: 295.7 KB
```

In [26]:
```
#colors colonnes
```

```
data['Color'].unique()
```

Out[26]:
```
array(['Silver', 'Black', 'White', 'Grey', 'Blue', 'Green', 'Red',
       'Sky blue', 'Orange', 'Yellow', 'Brown', 'Golden', 'Beige',
       'Carnelian red', 'Purple', 'Pink'], dtype=object)
```

Data Analyst

In [27]:
```
#TOP 5 fabricant par rapport prix
data.groupby('Manufacturer')['Price'].sum().sort_values(ascending=False).
```

Out[27]:
```
Manufacturer
HYUNDAI          83179077
TOYOTA           51574627
MERCEDES-BENZ    37980047
OPEL             29096058
BMW              21351409
Name: Price, dtype: int64
```

In [28]:
```
#TOP 5 Model les plus vendu
data.groupby("Model")['ID'].count().sort_values(ascending=False).head(5)
```

Out[28]:
```
Model
Prius      1069
Sonata     1067
Camry       929
Elantra     910
E 350       534
Name: ID, dtype: int64
```

In [29]:
```
# Nombre de vente de chaque type
data.groupby("Fuel type")['ID'].count().sort_values(ascending=False)
```

Out[29]:
```
Fuel type
Petrol            9944
Diesel            4001
Hybrid            3539
LPG                885
CNG                469
Plug-in Hybrid      85
Hydrogen             1
Name: ID, dtype: int64
```

In [30]:
```
#Combien de voiture de 0km est vendu
(data['Mileage (Km)']==0).sum()
```

Out[30]:
```
714
```

In [31]:
```
#combien de voiture en cuir
data['Leather interior'].value_counts()
```

Out[31]:
```
Yes    13731
No      5193
Name: Leather interior, dtype: int64
```

In [32]:
```
#Top 5 voiture vendu par category
data.groupby('Category')['ID'].count().sort_values(ascending=False).head(
```

Out[32]:
```
Category
Sedan        8600
Jeep         5378
Hatchback    2799
Minivan       633
Coupe         528
Name: ID, dtype: int64
```

In [33]:
```
#Top 5 années avec meilleur nombre de vente
data.groupby('Prod. year')['ID'].count().sort_values(ascending=False).hea
```

Out[33]:
```
Prod. year
2012    2131
2014    2090
2013    1913
2011    1582
2015    1527
Name: ID, dtype: int64
```

In [34]:
```
#Drive wheels
data['Drive wheels'].value_counts()
```

Out[34]:
```
Front    12695
4x4       3969
Rear      2260
Name: Drive wheels, dtype: int64
```

In [35]:
```
#top 5 couleurs vendu
data.groupby('Color')['ID'].count().sort_values(ascending=False).head(5)
```

Out[35]:
```
Color
Black     4944
White     4407
Silver    3729
Grey      2343
Blue      1376
Name: ID, dtype: int64
```

In [36]:
```
#Top 5 prix moyen par fabricant
data.groupby('Manufacturer')['Price'].mean().round(2).sort_values(ascendi
```

Out[36]:
```
Manufacturer
LAMBORGHINI    872946.00
BENTLEY        197574.50
OPEL            73474.89
FERRARI         66955.50
LAND ROVER      54807.19
Name: Price, dtype: float64
```

Sauvegarder nouveau fichier apres mise a jour pour Dashboard PowerBI

In [37]:
```
data.to_csv('fichier_modifier.csv',sep=',', index=False)
```

Data Scientist

In [38]:
```
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
from sklearn.preprocessing import LabelEncoder
```

In [39]:
```
cor=data.corr()
```

In [40]:
```
cor
```

Out[40]:

	ID	Price	Levy	Prod. year	Mileage (Km)	Cylinders	Ai
ID	1.000000	-0.000797	0.103537	0.072030	0.004225	-0.040617	-0.0
Price	-0.000797	1.000000	-0.003564	0.012689	-0.001763	0.007435	-0.0
Levy	0.103537	-0.003564	1.000000	0.363094	-0.021220	0.251415	0.1
Prod. year	0.072030	0.012689	0.363094	1.000000	-0.064058	-0.099443	0.2
Mileage (Km)	0.004225	-0.001763	-0.021220	-0.064058	1.000000	-0.001720	-0.0
Cylinders	-0.040617	0.007435	0.251415	-0.099443	-0.001720	1.000000	0.
Airbags	-0.022070	-0.012709	0.125028	0.235160	-0.009174	0.174112	1.0

In [41]:
```
sns.heatmap(cor,annot=True,fmt='.3f')
```

Out[41]:
```
<Axes: >
```

In [42]:
```
#Transform data
data_object=data.select_dtypes(include='object')
data_num=data.select_dtypes(exclude='object')
```

In [43]:
```
data_object
```

Out[43]:

	Manufacturer	Model	Category	Leather interior	Fuel type	Engine volume	Gear box type	Dr whe
0	LEXUS	RX 450	Jeep	Yes	Hybrid	3.5	Automatic	4
1	CHEVROLET	Equinox	Jeep	No	Petrol	3	Tiptronic	4
2	HONDA	FIT	Hatchback	No	Petrol	1.3	Variator	Fr
3	FORD	Escape	Jeep	Yes	Hybrid	2.5	Automatic	4
4	HONDA	FIT	Hatchback	Yes	Petrol	1.3	Automatic	Fr
...	...	...	...	...	...	...	...	
19232	MERCEDES-BENZ	CLK 200	Coupe	Yes	CNG	2.0 Turbo	Manual	R
19233	HYUNDAI	Sonata	Sedan	Yes	Petrol	2.4	Tiptronic	Fr
19234	HYUNDAI	Tucson	Jeep	Yes	Diesel	2	Automatic	Fr
19235	CHEVROLET	Captiva	Jeep	Yes	Diesel	2	Automatic	Fr
19236	HYUNDAI	Sonata	Sedan	Yes	Hybrid	2.4	Automatic	Fr

18924 rows × 11 columns

In [44]:
```
data_num
```

Out[44]:

	ID	Price	Levy	Prod. year	Mileage (Km)	Cylinders	Airbags
0	45654403	13328	1399	2010	186005	6	12
1	44731507	16621	1018	2011	192000	6	8
2	45774419	8467	-1	2006	200000	4	2
3	45769185	3607	862	2011	168966	4	0
4	45809263	11726	446	2014	91901	4	4
...	...	...	...	...	...	...	...
19232	45798355	8467	-1	1999	300000	4	5
19233	45778856	15681	831	2011	161600	4	8
19234	45804997	26108	836	2010	116365	4	4
19235	45793526	5331	1288	2007	51258	4	4
19236	45813273	470	753	2012	186923	4	12

18924 rows × 7 columns

In [45]:
```
la=LabelEncoder()
```

In [46]:
```
for i in range(0,data_object.shape[1]):
    data_object.iloc[:,i]= la.fit_transform(data_object.iloc[:,i])
```

In [47]:
```
data_object
```

Out[47]:

	Manufacturer	Model	Category	Leather interior	Fuel type	Engine volume	Gear box type	Drive wheels	Doc
0	32	1242	4	1	2	63	0	0	
1	8	658	4	0	5	56	2	0	
2	21	684	3	0	5	22	3	1	
3	16	661	4	1	2	46	0	0	
4	21	684	3	1	5	22	0	1	
...	...	...	...	...	...	...	...	...	
19232	36	385	1	1	0	37	1	2	
19233	23	1334	9	1	5	44	2	1	
19234	23	1442	4	1	1	36	0	1	
19235	8	456	4	1	1	36	0	1	
19236	23	1334	9	1	2	44	0	1	

18924 rows × 11 columns

In [48]:
```
data=pd.concat([data_object,data_num],axis=1)
```

In [49]:
```
data
```

Out[49]:

	Manufacturer	Model	Category	Leather interior	Fuel type	Engine volume	Gear box type	Drive wheels	Doc
0	32	1242	4	1	2	63	0	0	
1	8	658	4	0	5	56	2	0	
2	21	684	3	0	5	22	3	1	
3	16	661	4	1	2	46	0	0	
4	21	684	3	1	5	22	0	1	
...	...	...	...	...	...	...	...	...	
19232	36	385	1	1	0	37	1	2	
19233	23	1334	9	1	5	44	2	1	
19234	23	1442	4	1	1	36	0	1	
19235	8	456	4	1	1	36	0	1	
19236	23	1334	9	1	2	44	0	1	

18924 rows × 18 columns

In [50]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 18924 entries, 0 to 19236
Data columns (total 18 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Manufacturer      18924 non-null  int64
 1   Model             18924 non-null  int64
 2   Category          18924 non-null  int64
 3   Leather interior  18924 non-null  int64
 4   Fuel type         18924 non-null  int64
 5   Engine volume     18924 non-null  int64
 6   Gear box type     18924 non-null  int64
 7   Drive wheels      18924 non-null  int64
 8   Doors             18924 non-null  int64
 9   Wheel             18924 non-null  int64
 10  Color             18924 non-null  int64
 11  ID                18924 non-null  int64
 12  Price             18924 non-null  int64
 13  Levy              18924 non-null  int64
 14  Prod. year        18924 non-null  int64
 15  Mileage (Km)      18924 non-null  int64
 16  Cylinders         18924 non-null  int64
 17  Airbags           18924 non-null  int64
dtypes: int64(18)
memory usage: 2.7 MB
```

Models : Prediciton price

In [51]:
```
x=data.drop(['Price','Airbags','ID'],axis=1)
y=data['Price']
```

In [52]:
```
from sklearn.model_selection import train_test_split
```

In [53]:
```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,random
```

Import librairies Machine learning

In [54]:
```
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegre
from xgboost import XGBRegressor
from sklearn.svm import SVR
from sklearn.neighbors import KNeighborsRegressor


from sklearn.metrics import r2_score,mean_squared_error
import numpy as np
```

In [55]:
```
models_name=['LinearRegression','DecisionTreeRegressor','RandomForestRegr
R2=[]
RMSE=[]
```

In [56]:
```
#Fonction pour models
def models(model):
    model.fit(x_train,y_train)
    pre=model.predict(x_test)
    r2=r2_score(y_test,pre)
    R2.append(r2)
    rmse=np.sqrt(mean_squared_error(y_test,pre))
    RMSE.append(rmse)
    score=model.score(x_test,y_test)
    print(f"score est :{score}")
```

Model1 : Linear Regression

In [57]:
```
model1=LinearRegression()
model2=DecisionTreeRegressor()
model3=RandomForestRegressor()
model4=GradientBoostingRegressor()
model5=XGBRegressor()
model6=SVR()
model7=KNeighborsRegressor()
```

In [58]:
```
models(model1)
models(model2)
models(model3)
models(model4)
models(model5)
models(model6)
models(model7)
```

```
score est :-0.10206908174723228
score est :0.31669618292639623
score est :-9.851033132487494
score est :-22.974616253613444
score est :0.5964197074906046
score est :-0.048129020371816544
score est :0.11755191744736904
```

In [59]:
```
pd.DataFrame({'models_name':models_name,'R2_score':R2,'RMSE':RMSE})
```

Out[59]:

	models_name	R2_score	RMSE
0	LinearRegression	-0.102069	19533.750355
1	DecisionTreeRegressor	0.316696	15381.126614
2	RandomForestRegressor	-9.851033	61293.831282
3	GradientBoostingRegressor	-22.974616	91108.113887
4	XGBRegressor	0.596420	11820.781757
5	SVR	-0.048129	19049.720002
6	KNeighborsRegressor	0.117552	17479.371213

In [ ]:
```

```

SUPERSTORE SALES ANALYSIS DASHBOARD

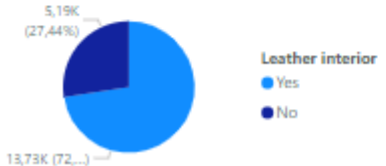
Sales Count	Model Count	Manufacturer Count	Revenue
18,924K	1,511K	65	352M

Filter by Year Production

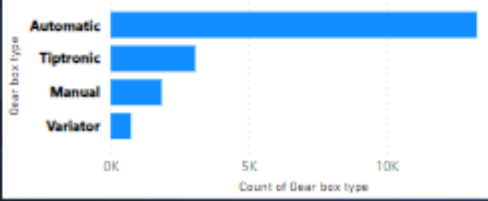
1939 2020

List of Models	List of Manufacturer	List of Category	List of Color
09-Mar	ACURA	Cabriolet	Beige
100	ALFA ROMEO	Coupe	Black
100 NX	ASTON MARTIN	Goods wagon	Blue
1000	AUDI	Hatchback	Brown
1111	BENTLEY	Jeep	Carnelian red
114	BMW	Limousine	Golden

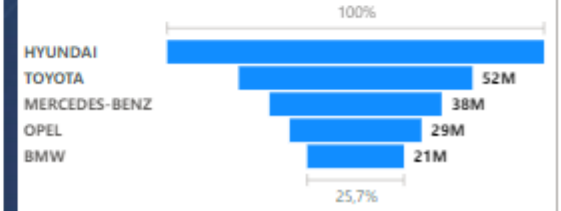
Leather vs. Non-leather Car Production



Count of GEAR BOX TYPE



TOP 5 Manufacturer Revenu



Detail by Manufacturer

Manufacturer

Manufacturer

- [ ] ACURA
- [ ] ALFA ROMEO
- [ ] ASTON MARTIN
- [ ] AUDI
- [ ] BENTLEY
- [x] BMW
- [ ] BUICK
- [ ] CADILLAC
- [ ] CHEVROLET
- [ ] CHRYSLER
- [ ] CITROEN
- [ ] DAEWOO
- [ ] DAIHATSU
- [ ] DODGE
- [ ] FERRARI
- [ ] FIAT
- [ ] FORD
- [ ] GAZ
- [ ] GMC
- [ ] GREATWALL
- [ ] HAVAL

Sales Count	Model Count	Revenue	Count Category
1,036K	155	21M	7

Count Model

Model	Count of Model
X5	351
328	75
528	58
535	49
X6	44
320	39
318	34
530	31
525	29
325	21
335	21
Total	1036

Count Color Sales

Color	Count of Color
Black	429
White	175
Silver	127
Grey	119
Blue	99
Green	24
Brown	17
Red	16
Beige	9
Sky blue	7
Golden	5
Total	1036

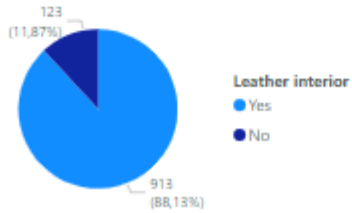
Revenue by Category

Category	Sum of Price
Jeep	9634K
Sedan	9353K
Coupe	1807K
Hatchback	279K
Cabriolet	179K
Universal	100K
Limousine	0K
Total	21351K

Revenue by Model

Model	Sum of Price
114	14K
116	9K
118	19K
118 2,0	20K
118 M-sport LCI	22K
120	9K
128	21K
128 M tech	22K
130	13K
135	3K
225	25K
Total	21351K

Leather vs. Non-leather Car Sales



Revenue by Year