Cars Data Analyse and Visualization

Import Libraries

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import statistics
import seaborn as sns
import plotly.figure_factory as ff
import plotly.graph_objs as go
from plotly.offline import download_plotlyjs, init_notebook_mode, plot, iplot
import plotly.express as px
import requests
```

Read data

In [3]:
```
df_hyundi = pd.read_csv('C:/Users/Downloads/hyundi.csv')
df_hyundi.head()
```

Out[3]:

	model	year	price	transmission	mileage	fuelType	tax(£)	mpg	engineSize
0	I20	2017	7999	Manual	17307	Petrol	145	58.9	1.2
1	Tucson	2016	14499	Automatic	25233	Diesel	235	43.5	2.0
2	Tucson	2016	11399	Manual	37877	Diesel	30	61.7	1.7
3	I10	2016	6499	Manual	23789	Petrol	20	60.1	1.0
4	IX35	2015	10199	Manual	33177	Diesel	160	51.4	2.0

Columns deleted

In [4]:
```
df_hyundi = df_hyundi.drop(columns={'tax(£)': 'tax'})
df_hyundi.head()
```

Out[4]:

	model	year	price	transmission	mileage	fuelType	mpg	engineSize
0	I20	2017	7999	Manual	17307	Petrol	58.9	1.2
1	Tucson	2016	14499	Automatic	25233	Diesel	43.5	2.0
2	Tucson	2016	11399	Manual	37877	Diesel	61.7	1.7
3	I10	2016	6499	Manual	23789	Petrol	60.1	1.0
4	IX35	2015	10199	Manual	33177	Diesel	51.4	2.0

Merging all data sets to one big full_data dataset:

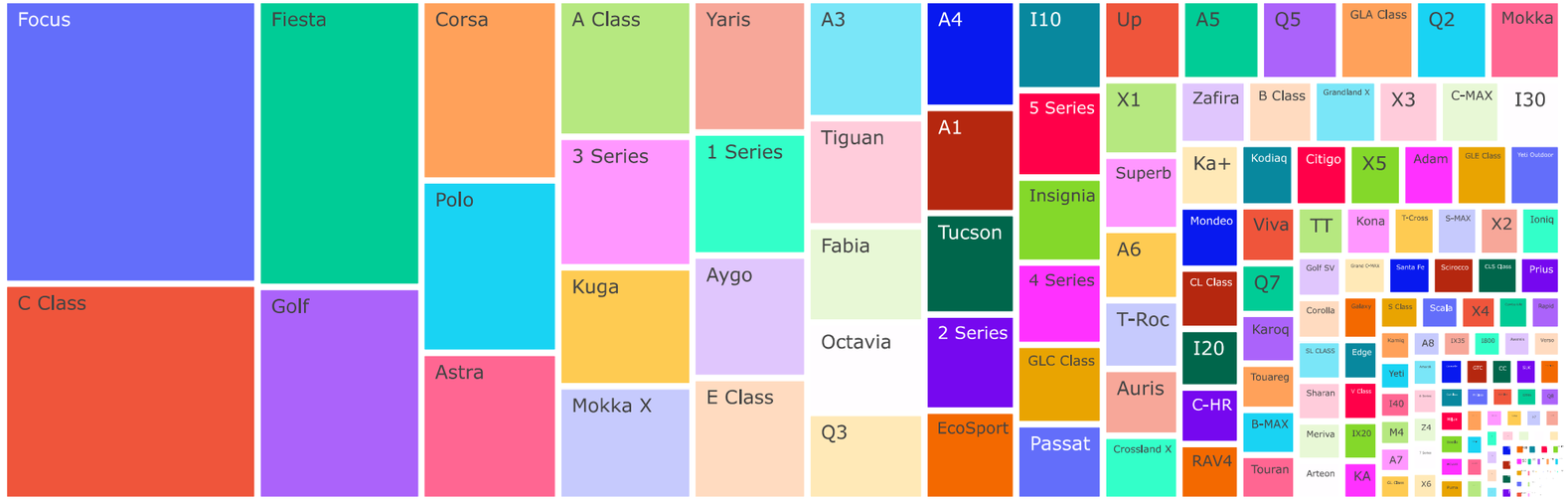
In [8]:
```
# This will get the names of the files in the dataset's folder
files = [file for file in os.listdir('C:/Users/Desktop/cars dataset')]
```

```
full_data = pd.DataFrame()

for file in files:
    if file in ['hyundi.csv', 'unclean focus.csv', 'unclean cclass.csv']:
        continue
    df = pd.read_csv('C:/Users/Desktop/cars dataset/'+file)
    full_data = pd.concat([full_data, df])

full_data = pd.concat([full_data, df_hyundi])
print(full_data.head())
```

```
  model  year  price transmission  mileage fuelType    tax   mpg  engineSize
0    A1  2017  12500       Manual    15735   Petrol  150.0  55.4         1.4
1    A6  2016  16500    Automatic    36203   Diesel   20.0  64.2         2.0
2    A1  2016  11000       Manual    29946   Petrol   30.0  55.4         1.4
3    A4  2017  16800    Automatic    25952   Diesel  145.0  67.3         2.0
4    A3  2019  17300       Manual     1998   Petrol  145.0  49.6         1.0
```

In [9]:
```
model_count=full_data
ax2= px.treemap(model_count,path=["model"],title="Popularity of the bought car models:")
ax2.show()
```

Popularity of the bought car models:



Preliminary statistical analysis:

In [10]:
```
print(full_data.describe())
```

```
                year          price        mileage           tax  \
count  108540.000000  108540.000000  108540.000000  94327.000000
mean     2017.098028   16890.124046   23025.928469    120.256183
std         2.130057    9756.266820   21176.423684     63.404805
min      1970.000000     450.000000       1.000000      0.000000
25%      2016.000000   10229.500000    7491.750000    125.000000
50%      2017.000000   14698.000000   17265.000000    145.000000
75%      2019.000000   20940.000000   32236.000000    145.000000
max      2060.000000  159999.000000  323000.000000    580.000000

                mpg     engineSize
count  99187.000000  108540.000000
mean      55.166825       1.661644
std       16.138522       0.557058
min        0.300000       0.000000
25%       47.100000       1.200000
50%       54.300000       1.600000
75%       62.800000       2.000000
max      470.800000       6.600000
```

"shape"

In [11]:
```
full_data=full_data[full_data.year != 2060]
```

In [12]:
```
print(full_data.shape)
```

```
(108539, 9)
```

"columns"

In [13]:
```
print(full_data.columns)
```

```
Index(['model', 'year', 'price', 'transmission', 'mileage', 'fuelType', 'tax',
       'mpg', 'engineSize'],
      dtype='object')
```

"info"

In [14]:
```
print(full_data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 108539 entries, 0 to 4859
Data columns (total 9 columns):
 #   Column        Non-Null Count   Dtype  
---  ------        --------------   -----  
 0   model         108539 non-null  object 
 1   year          108539 non-null  int64  
 2   price         108539 non-null  int64  
 3   transmission  108539 non-null  object 
 4   mileage       108539 non-null  int64  
 5   fuelType      108539 non-null  object 
 6   tax           94326 non-null   float64
 7   mpg           99186 non-null   float64
 8   engineSize    108539 non-null  float64
dtypes: float64(3), int64(3), object(3)
memory usage: 8.3+ MB
None
```

"isna"

In [15]:
```
full_data.isna().any()
```

Out[15]:
```
model           False
year            False
price           False
transmission    False
mileage         False
fuelType        False
tax              True
mpg              True
engineSize      False
dtype: bool
```

Calculating which models are most popular:

Calculate total amount of the sales for each model:

Importing Cyberpunk visual style

In [16]:
```
!pip install mplcyberpunk
```

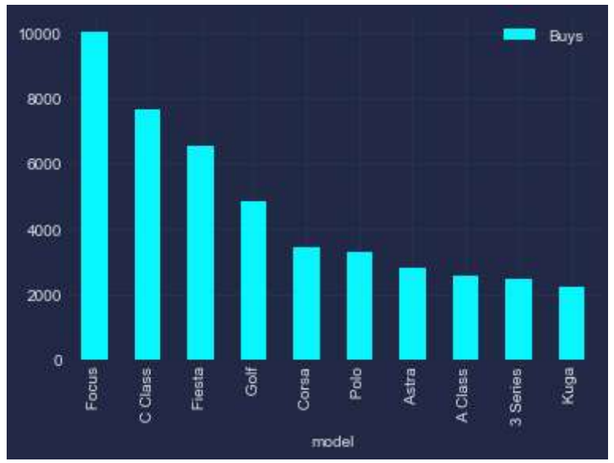
```
Collecting mplcyberpunk
  Downloading mplcyberpunk-0.7.0-py3-none-any.whl (6.3 kB)
Requirement already satisfied: matplotlib in c:\users\zizhe\new folder\lib\site-packages (from mplcyberpunk) (3.5.1)
Requirement already satisfied: packaging>=20.0 in c:\users\zizhe\new folder\lib\site-packages (from matplotlib->mplcyberpunk) (21.3)
Requirement already satisfied: pyparsing>=2.2.1 in c:\users\zizhe\new folder\lib\site-packages (from matplotlib->mplcyberpunk) (3.0.4)
WARNING: Ignoring invalid distribution -rotobuf (c:\users\zizhe\new folder\lib\site-packages)
WARNING: Ignoring invalid distribution -rotobuf (c:\users\zizhe\new folder\lib\site-packages)
WARNING: Ignoring invalid distribution -rotobuf (c:\users\zizhe\new folder\lib\site-packages)
WARNING: Ignoring invalid distribution -rotobuf (c:\users\zizhe\new folder\lib\site-packages)
WARNING: Ignoring invalid distribution -rotobuf (c:\users\zizhe\new folder\lib\site-packages)
WARNING: Ignoring invalid distribution -rotobuf (c:\users\zizhe\new folder\lib\site-packages)
WARNING: Ignoring invalid distribution -rotobuf (c:\users\zizhe\new folder\lib\site-packages)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\zizhe\new folder\lib\site-packages (from matplotlib->mplcyberpunk) (2.8.2)
Requirement already satisfied: numpy>=1.17 in c:\users\zizhe\appdata\roaming\python\python39\site-packages (from matplotlib->mplcyberpunk) (1.25.2)
Requirement already satisfied: pillow>=6.2.0 in c:\users\zizhe\new folder\lib\site-packages (from matplotlib->mplcyberpunk) (9.5.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\zizhe\new folder\lib\site-packages (from matplotlib->mplcyberpunk) (4.25.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\zizhe\new folder\lib\site-packages (from matplotlib->mplcyberpunk) (1.3.2)
Requirement already satisfied: cycler>=0.10 in c:\users\zizhe\new folder\lib\site-packages (from matplotlib->mplcyberpunk) (0.11.0)
Requirement already satisfied: six>=1.5 in c:\users\zizhe\new folder\lib\site-packages (from python-dateutil>=2.7->matplotlib->mplcyberpunk) (1.16.0)
Installing collected packages: mplcyberpunk
Successfully installed mplcyberpunk-0.7.0
```

In [17]:
```
import mplcyberpunk
```

In [18]:
```
gry=full_data
model_buys= gry.groupby('model')['model'].count()
model_buys = pd.DataFrame(model_buys)
model_buys.columns = ['Buys']
model_buys.sort_values(by=['Buys'], inplace=True, ascending=False)
model_buys = model_buys.head(10)
print(model_buys.head(20))
plt.style.use("cyberpunk")
model_buys.plot.bar()
```

```
            Buys
model
 Focus     10042
 C Class    7646
 Fiesta     6556
 Golf       4863
 Corsa      3441
 Polo       3287
 Astra      2805
 A Class    2561
 3 Series   2443
 Kuga       2225
```

Out[18]: <AxesSubplot:xlabel='model'>



In the UK, the most popular car models include the Focus, C Class, and Fiesta. It's evident that Ford cars reign as the kings of the roads.

Checking the year from which cars are the most popular:

In [19]:
```
gry=full_data
model_buys1= gry.groupby('year')['model'].count()
model_buys1 = pd.DataFrame(model_buys1)
model_buys1.columns = ['Buys']
model_buys1.sort_values(by=['Buys'], inplace=True, ascending=False)
model_buys1 = model_buys1.head(10)
print(model_buys1.head(20))
```

```
      Buys
year
2019  29955
2017  23686
2016  16948
2018  15092
2015   8359
2020   4657
2014   4377
2013   2818
2012    702
2011    477
```

In [20]:
```
plt.style.use("cyberpunk")
model_buys1.plot.bar()
```

Out[20]:
```
<AxesSubplot:xlabel='year'>
```

In the UK, it's evident that the most popular car year is 2019. Interestingly, people seem to prefer purchasing cars that are within the last 4 years over brand-new vehicles.

Checking the average price of cars regarding their year of registration

In [21]:
```
model_buys2=full_data
model_buys2=model_buys2[["price","year"]].groupby('year').mean()
model_buys2.sort_values(by=['price'], inplace=True, ascending=False)
model_buys2 = model_buys2.head(10)
print(model_buys2.head(20))
```

```
             price
year
2020  28517.838737
2019  23535.764814
1970  17747.000000
2018  16310.332229
2017  14463.352022
2016  13374.278322
2015  11655.549587
2014   9998.366690
2013   8539.841732
2012   7520.623932
```

In [22]:
```
plt.figure(figsize=(14,7))

# Add title
plt.title("Average Money Spend for a model, by Year")

# Heatmap showing average arrival delay for each airline by month
sns.heatmap(data=model_buys2, annot=True)
# Add label for horizontal axis
plt.xlabel("Model price")
```

Out[22]:
```
Text(0.5, 46.5, 'Model price')
```

While one might expect newer cars to have the highest prices per unit, it's intriguing to note that classic cars from the 1970s are priced higher than cars from 2018 and 2017.

Let's check which transmission is more popular

In [23]:
```
model_trans=gry
model_trans= model_trans.groupby('transmission')['model'].count()
print(model_trans.head(20))
```

```
transmission
Automatic    22318
Manual       61308
Other           10
Semi-Auto    24903
Name: model, dtype: int64
```

As we can see the most of the bought cars have manual gearbox

Visualization of the transmissions:

In [24]:
```
model_trans=full_data
model_trans=full_data
model_trans=model_trans[model_trans.year != 1970]
model_trans=model_trans[model_trans.year != 1991]

model_trans3 = model_trans[model_trans.transmission == "Automatic"]
model_trans3= model_trans.groupby('year')['model'].count()

print(model_trans3.head(50))
```

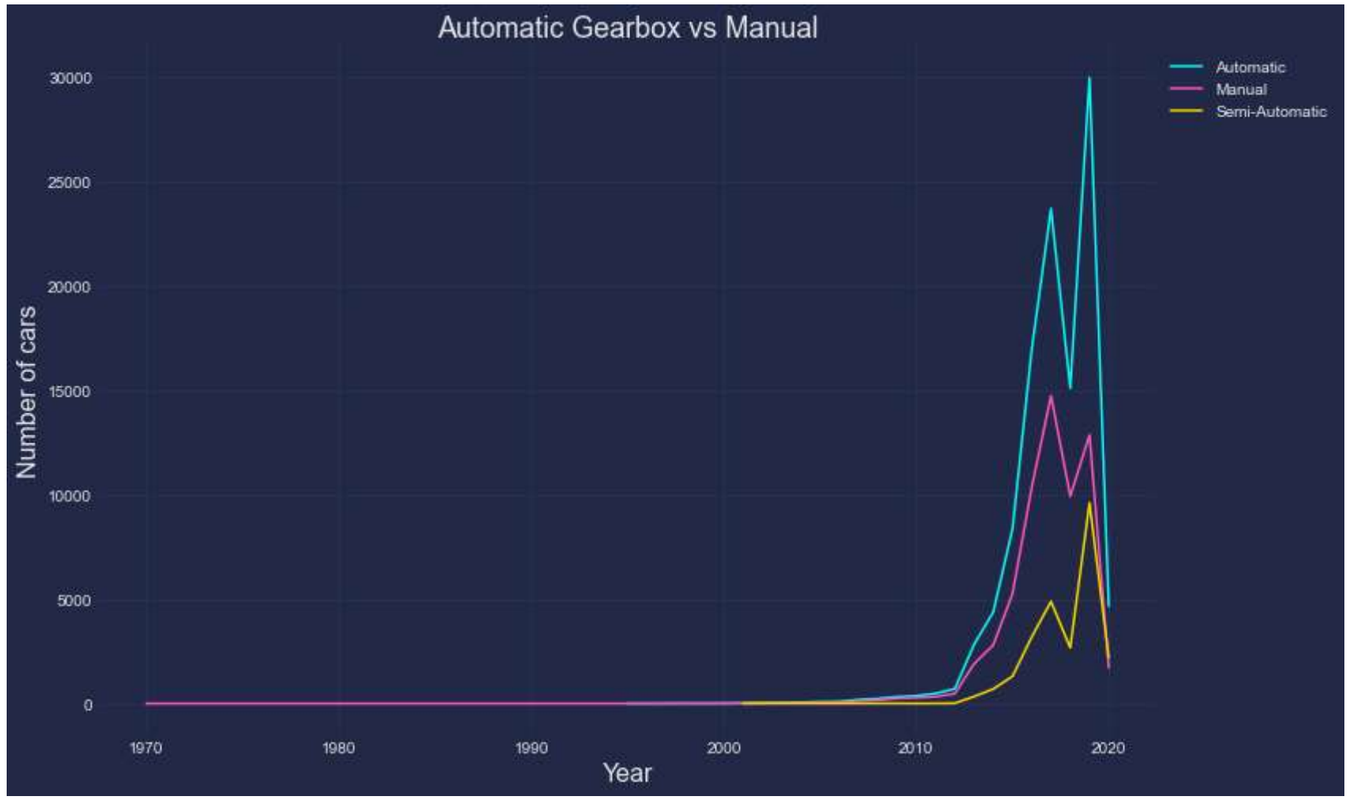
```
year
1995        1
1996        2
1997        4
1998        9
1999        6
2000        9
2001       20
2002       36
2003       39
2004       55
2005       84
2006       92
2007      185
2008      233
2009      321
2010      369
2011      477
2012      702
2013     2818
2014     4377
2015     8359
2016    16948
2017    23686
2018    15092
2019    29955
2020     4657
Name: model, dtype: int64
```

In [25]:
```
model_trans2=full_data
model_trans2 = model_trans2[model_trans2.transmission == "Manual"]
model_trans2= model_trans2.groupby('year')['model'].count()
```

In [26]:
```
model_trans4=full_data
model_trans4 = model_trans4[model_trans4.transmission == "Semi-Auto"]
model_trans4= model_trans4.groupby('year')['model'].count()
```

Manual vs Automatic:

In [27]:
```
plt.style.use("cyberpunk")
# plot out the time series
model_trans3.plot(kind = 'line', label = 'Automatic', figsize = (12, 8))
model_trans2.plot(kind = 'line', label = 'Manual', figsize = (12, 8))
model_trans4.plot(kind = 'line', label = 'Semi-Automatic', figsize = (12, 8))
plt.title('Automatic Gearbox vs Manual', fontsize = 18)
plt.xlabel('Year', fontsize = 16)
plt.ylabel('Number of cars', fontsize = 16)
plt.legend(loc='upper left', bbox_to_anchor=(1,1), ncol=1);
```



It's evident that the majority of purchased cars currently have a manual gearbox. However, there's a noticeable shift in this trend as an increasing number of people are showing interest in cars equipped with automatic gearboxes.

Let's check which fuel type is more popular

In [28]:
```
model_fuel=gry
model_fuel= model_fuel.groupby('fuelType')['model'].count()
print(model_fuel.head(20))
```

```
fuelType
Diesel      45177
Electric        6
Hybrid       3229
Other         253
Petrol      59874
Name: model, dtype: int64
```

As we can see the most of the bought cars are Diesel

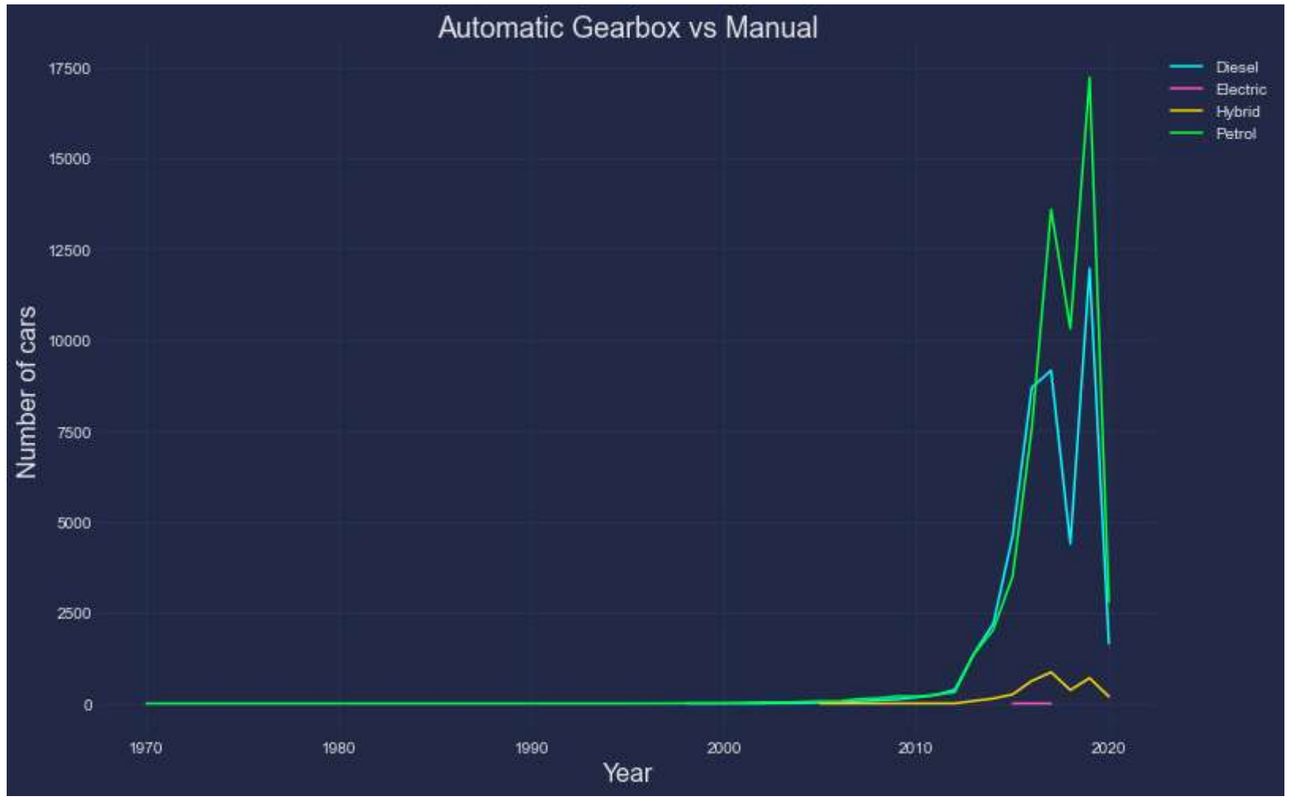
Visualization of the fuel types:

In [29]:
```
model_fuel=full_data
model_fuel1=full_data
model_fuel2=full_data
model_fuel3=full_data
model_fuel4=full_data
```

```
model_fuel=full_data
model_fuel=model_fuel[model_fuel.year != 1970]
model_fuel=model_fuel[model_fuel.year != 1991]

model_fuel1=model_fuel1[model_fuel1.year != 1970]
model_fuel1=model_fuel1[model_fuel1.year != 1991]

model_fuel1 = model_fuel1[model_fuel1.fuelType == "Diesel"]
model_fuel1= model_fuel1.groupby('year')['model'].count()

model_fuel2 = model_fuel2[model_fuel2.fuelType == "Electric"]
model_fuel2= model_fuel2.groupby('year')['model'].count()

model_fuel3 = model_fuel3[model_fuel3.fuelType == "Hybrid"]
model_fuel3= model_fuel3.groupby('year')['model'].count()

model_fuel4 = model_fuel4[model_fuel4.fuelType == "Petrol"]
model_fuel4= model_fuel4.groupby('year')['model'].count()
```

In [30]:

```
plt.style.use("cyberpunk")
# plot out the time series
model_fuel1.plot(kind = 'line', label = "Diesel", figsize = (12, 8))
model_fuel2.plot(kind = 'line', label = "Electric", figsize = (12, 8))
model_fuel3.plot(kind = 'line', label = "Hybrid", figsize = (12, 8))
model_fuel4.plot(kind = 'line', label = "Petrol", figsize = (12, 8))
plt.title('Automatic Gearbox vs Manual', fontsize = 18)
plt.xlabel('Year', fontsize = 16)
plt.ylabel('Number of cars', fontsize = 16)
plt.legend(loc='upper left', bbox_to_anchor=(1,1), ncol=1);
```



It's evident that:

Over the past year, there has been a noticeable increase in the number of hybrid cars, alongside the emergence of some electric vehicles.

Furthermore, there's a notable rise in the quantity of cars equipped with diesel engines. This surge might be attributed to the development of new high-performance diesel engines renowned for their suitability for long rides.

Investigation over the price changes:

In [34]:
```
from plotly.subplots import make_subplots
import plotly.graph_objects as go
```

In [35]:
```
plt.style.use("cyberpunk")
dfer= full_data
dfmer= full_data
dfwil= full_data
drai=full_data
dfer =dfer.sort_values('mpg', ascending=False)
dfmer =dfmer.sort_values('engineSize', ascending=False)
dfwil =dfwil.sort_values('tax', ascending=False)
drai =drai.sort_values('mileage', ascending=False)
fig = make_subplots(rows=2, cols=2)
fig.add_trace(go.Scatter(x=dfer['mpg'], y=dfer['price'], name="Miles Per Galon"),row=1, col=1)
fig.add_trace(go.Scatter(x=dfmer['engineSize'], y=dfmer['price'], name="Engine Size"),row=1, col=2)
fig.add_trace(go.Scatter(x=dfwil['tax'], y=dfwil['price'], name="Tax"),row=2, col=1)
fig.add_trace(go.Scatter(x=drai['mileage'], y=dfwil["price"], name="Mileage"),row=2, col=2)
```

```
fig.update_layout(height=600, width=650, title_text="What the price depends on?")
fig.show()
plt.style.use("cyberpunk")
```

It's noticeable that:

The majority of cars fall within the first interval of up to 100 miles per gallon.

The highest price tends to correspond to the smallest amount of miles per gallon, although it's essential to consider that this correlation is also linked to the engine size.

Upon checking the tax amounts, a clear correlation isn't evident with the year, mpg, or engine size. This is due to the tax being contingent on the carbon emissions of the car in the UK.

Additionally, a smaller amount of mileage also contributes to a higher price.

Summary

In the UK, the most popular car models are the Ford Focus, Mercedes C Class, and Ford Fiesta. It's evident that Ford is the favorite car brand in the United Kingdom.

Most of the cars sold were purchased within the last 3 to 4 years. The majority of these cars are equipped with a manual gearbox, and petrol stands as the most commonly used fuel type.

Determining the price of a car depends on numerous factors, making it challenging to pinpoint the primary reason for significant price changes. However, it's noticeable that prices tend to be higher for cars that consume smaller amounts of fuel per gallon and have lower mileage. It's worth noting that taxation in the UK is not solely determined by the car's price; it also varies based on CO2 emissions.