jupyter autos Last Checkpoint: hace un día (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Para salir de la pantalla completa, mueve el mouse hacia la parte superior de la pantalla o presiona F11

Run Code

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import plotly.graph_objects as go

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

from sklearn.ensemble import RandomForestRegressor
from sklearn.pipeline import make_pipeline

import math
from sklearn.metrics import r2_score, mean_squared_error

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

import warnings
warnings.filterwarnings('ignore')
```

In [3]:
```
cars_data = pd.read_csv('data.csv')
```

In [4]:
```
cars_data.head()
```

Out[4]:

	Make	Model	Year	Engine Fuel Type	Engine HP	Engine Cylinders	Transmission Type	Driven_Wheels	Number of Doors	Market Category	Vehicle Size	Vehicle Style	highway MPG	city mpg	Popularity	MSF
0	BMW	1 Series M	2011	premium unleaded (required)	335.0	6.0	MANUAL	rear wheel drive	2.0	Factory Tuner,Luxury,High-Performance	Compact	Coupe	26	19	3916	461
1	BMW	1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact	Convertible	28	19	3916	406
2	BMW	1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,High-Performance	Compact	Coupe	28	20	3916	363

jupyter autos Last Checkpoint: hace un día (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

```
warnings.filterwarnings('ignore')
```

In [3]:
```
cars_data = pd.read_csv('data.csv')
```

In [4]:
```
cars_data.head()
```

Out[4]:

	Make	Model	Year	Engine Fuel Type	Engine HP	Engine Cylinders	Transmission Type	Driven_Wheels	Number of Doors	Market Category	Vehicle Size	Vehicle Style	highway MPG	city mpg	Popularity	MSF
0	BMW	1 Series M	2011	premium unleaded (required)	335.0	6.0	MANUAL	rear wheel drive	2.0	Factory Tuner,Luxury,High-Performance	Compact	Coupe	26	19	3916	461
1	BMW	1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact	Convertible	28	19	3916	406
2	BMW	1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,High-Performance	Compact	Coupe	28	20	3916	363
3	BMW	1 Series	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact	Coupe	28	18	3916	294
4	BMW	1 Series	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury	Compact	Convertible	28	18	3916	345

In [5]:
```
cars_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11914 entries, 0 to 11913
Data columns (total 16 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Make               11914 non-null  object
 1   Model              11914 non-null  object
 2   Year               11914 non-null  int64
 3   Engine Fuel Type   11911 non-null  object
 4   Engine HP          11845 non-null  float64
 5   Engine Cylinders   11884 non-null  float64
 6   Transmission Type  11914 non-null  object
 7   Driven_Wheels      11914 non-null  object
 8   Number of Doors    11908 non-null  float64
 9   Market Category    8172 non-null   object
```

jupyter autos Last Checkpoint: hace un día (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11914 entries, 0 to 11913
Data columns (total 16 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Make               11914 non-null  object
 1   Model              11914 non-null  object
 2   Year               11914 non-null  int64
 3   Engine Fuel Type   11911 non-null  object
 4   Engine HP          11845 non-null  float64
 5   Engine Cylinders   11884 non-null  float64
 6   Transmission Type  11914 non-null  object
 7   Driven_Wheels      11914 non-null  object
 8   Number of Doors    11908 non-null  float64
 9   Market Category    8172 non-null   object
 10  Vehicle Size       11914 non-null  object
 11  Vehicle Style      11914 non-null  object
 12  highway MPG        11914 non-null  int64
 13  city mpg           11914 non-null  int64
 14  Popularity         11914 non-null  int64
 15  MSRP               11914 non-null  int64
dtypes: float64(3), int64(5), object(8)
memory usage: 1.5+ MB
```

In [6]:
```
cars_data.describe()
```

Out[6]:

	Year	Engine HP	Engine Cylinders	Number of Doors	highway MPG	city mpg	Popularity	MSRP
count	11914.000000	11845.00000	11884.000000	11908.000000	11914.000000	11914.000000	11914.000000	1.191400e+04
mean	2010.384338	249.38607	5.628829	3.436093	26.637485	19.733255	1554.911197	4.059474e+04
std	7.579740	109.19187	1.780559	0.881315	8.863001	8.987798	1441.855347	6.010910e+04
min	1990.000000	55.00000	0.000000	2.000000	12.000000	7.000000	2.000000	2.000000e+03
25%	2007.000000	170.00000	4.000000	2.000000	22.000000	16.000000	549.000000	2.100000e+04
50%	2015.000000	227.00000	6.000000	4.000000	26.000000	18.000000	1385.000000	2.999500e+04
75%	2016.000000	300.00000	6.000000	4.000000	30.000000	22.000000	2009.000000	4.223125e+04
max	2017.000000	1001.00000	16.000000	4.000000	354.000000	137.000000	5657.000000	2.065902e+06

In [7]:
```
cars_data.columns = cars_data.columns.str.lower().str.replace(" ", "_")
cars_data.rename(columns = {'engine_fuel_type' : 'fuel_type', 'engine_hp' : 'hp', 'engine_cylinders' : 'cylinders', 'transmissior
```

jupyter autos Last Checkpoint: hace un día (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

Run Code

50%	2015.000000	227.00000	6.000000	4.000000	26.000000	18.000000	1385.000000	2.999500e+04
75%	2016.000000	300.00000	6.000000	4.000000	30.000000	22.000000	2009.000000	4.223125e+04
max	2017.000000	1001.00000	16.000000	4.000000	354.000000	137.000000	5657.000000	2.065902e+06

In [7]:
```
cars_data.columns = cars_data.columns.str.lower().str.replace(" ", "_")
cars_data.rename(columns = {'engine_fuel_type' : 'fuel_type', 'engine_hp' : 'hp', 'engine_cylinders' : 'cylinders', 'transmissior
```

In [8]:
```
print('Number of duplicates are : ', cars_data.duplicated().sum())
cars_data = cars_data.drop_duplicates()
```

```
Number of duplicates are :  715
```

In [9]:
```
print('Number of missing values in each columns are below : ')
print(cars_data.isnull().sum())
```

```
Number of missing values in each columns are below :
make              0
model             0
year              0
fuel_type         3
hp               69
cylinders        30
transmission      0
drive             0
doors             6
market         3376
size              0
style             0
highway_mpg       0
city_mpg          0
popularity        0
price             0
dtype: int64
```

In [10]:
```
cars_data.drop('market', axis = 1, inplace = True)
```

In [11]:
```
null_values = cars_data[cars_data.isnull().any(axis = 1)]
null_values
```

Out[11]:

make	model	year	fuel_type	hp	cylinders	transmission	drive	doors	size	style	highway_mpg	city_mpg	popularity	price

jupyter autos Last Checkpoint: hace un día (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

```
highway_mpg        0
city_mpg           0
popularity         0
price              0
dtype: int64
```

In [10]:
```
cars_data.drop('market', axis = 1, inplace = True)
```

In [11]:
```
null_values = cars_data[cars_data.isnull().any(axis = 1)]
null_values
```

Out[11]:

	make	model	year	fuel_type	hp	cylinders	transmission	drive	doors	size	style	highway_mpg	city_mpg	popularity	price
539	FIAT	500e	2015	electric	NaN	0.0	DIRECT_DRIVE	front wheel drive	2.0	Compact	2dr Hatchback	108	122	819	31800
540	FIAT	500e	2016	electric	NaN	0.0	DIRECT_DRIVE	front wheel drive	2.0	Compact	2dr Hatchback	103	121	819	31800
541	FIAT	500e	2017	electric	NaN	0.0	DIRECT_DRIVE	front wheel drive	2.0	Compact	2dr Hatchback	103	121	819	31800
1983	Chevrolet	Bolt EV	2017	electric	200.0	NaN	DIRECT_DRIVE	front wheel drive	4.0	Compact	4dr Hatchback	110	128	1385	40905
1984	Chevrolet	Bolt EV	2017	electric	200.0	NaN	DIRECT_DRIVE	front wheel drive	4.0	Compact	4dr Hatchback	110	128	1385	36620
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
9853	Kia	Soul EV	2016	electric	NaN	0.0	DIRECT_DRIVE	front wheel drive	4.0	Compact	Wagon	92	120	1720	31950
9854	Kia	Soul EV	2016	electric	NaN	0.0	DIRECT_DRIVE	front wheel drive	4.0	Compact	Wagon	92	120	1720	35950
11321	Suzuki	Verona	2004	NaN	155.0	6.0	AUTOMATIC	front wheel drive	4.0	Midsize	Sedan	25	17	481	17199
11322	Suzuki	Verona	2004	NaN	155.0	6.0	AUTOMATIC	front wheel drive	4.0	Midsize	Sedan	25	17	481	20199
11323	Suzuki	Verona	2004	NaN	155.0	6.0	AUTOMATIC	front wheel drive	4.0	Midsize	Sedan	25	17	481	18499

102 rows × 15 columns

In [12]:
```
cars_data['fuel_type'] = cars_data['fuel_type'].fillna('regular unleaded')

cars_data['hp'] = cars_data['hp'].fillna(0)
```

jupyter autos Last Checkpoint: hace un día (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

In [12]:
```
cars_data['fuel_type'] = cars_data['fuel_type'].fillna('regular unleaded')

cars_data['hp'] = cars_data['hp'].fillna(0)

cars_data['cylinders'] = cars_data['cylinders'].fillna(0)

cars_data['doors'] = cars_data['doors'].fillna(cars_data['doors'].mean())
```

In [13]:
```
num_col = cars_data.select_dtypes(include = [np.number])
cat_col = cars_data.select_dtypes(exclude = [np.number])
```

In [14]:
```
for col in cat_col:
    print(col)
    print(cars_data[col].unique())
    print(cars_data[col].nunique())
    print('\n', "====================================", '\n')
```

```
make
['BMW' 'Audi' 'FIAT' 'Mercedes-Benz' 'Chrysler' 'Nissan' 'Volvo' 'Mazda'
 'Mitsubishi' 'Ferrari' 'Alfa Romeo' 'Toyota' 'McLaren' 'Maybach'
 'Pontiac' 'Porsche' 'Saab' 'GMC' 'Hyundai' 'Plymouth' 'Honda'
 'Oldsmobile' 'Suzuki' 'Ford' 'Cadillac' 'Kia' 'Bentley' 'Chevrolet'
 'Dodge' 'Lamborghini' 'Lincoln' 'Subaru' 'Volkswagen' 'Spyker' 'Buick'
 'Acura' 'Rolls-Royce' 'Maserati' 'Lexus' 'Aston Martin' 'Land Rover'
 'Lotus' 'Infiniti' 'Scion' 'Genesis' 'HUMMER' 'Tesla' 'Bugatti']
48

 ====================================

model
['1 Series M' '1 Series' '100' '124 Spider' '190-Class' '2 Series' '200'
 '200SX' '240SX' '240' '2' '3 Series Gran Turismo' '3 Series' '300-Class'
 '3000GT' '300' '300M' '300ZX' '323' '350-Class' '350Z' '360' '370Z' '3'
 '4 Series Gran Coupe' '4 Series' '400-Class' '420-Class' '456M'
 '458 Italia' '4C' '4Runner' '5 Series Gran Turismo' '5 Series'
 '500-Class' '500e' '500' '500L' '500X' '550' '560-Class' '570S' '575M'
```

In [15]:
```
cars_data.drop(cars_data[cars_data['transmission']=='UNKNOWN'].index, axis='index', inplace = True)
```

In [16]:
```
for i in num_col:
    fig = px.box(cars_data, x = cars_data[i])
```

jupyter autos Last Checkpoint: hace un día (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

Run Code

```
'458 Italia' '4C' '4Runner' '5 Series Gran Turismo' '5 Series'
'500-Class' '500e' '500' '500L' '500X' '550' '560-Class' '570S' '575M'
```

In [15]:
```
cars_data.drop(cars_data[cars_data['transmission']=='UNKNOWN'].index, axis='index', inplace = True)
```

In [16]:
```
for i in num_col:
    fig = px.box(cars_data, x = cars_data[i])
    fig.update_traces(fillcolor = '#C9A26B')
    fig.show()
```

In [17]:
```
s1 = cars_data.shape
clean = cars_data[['hp', 'cylinders', 'highway_mpg', 'city_mpg', 'price']]
for i in clean.columns:
    qt1 = cars_data[i].quantile(0.25)
    qt3 = cars_data[i].quantile(0.75)
    iqr =  qt3 - qt1
    lower = qt1-(1.5*iqr)
    upper = qt3+(1.5*iqr)
    min_in = cars_data[cars_data[i]<lower].index
    max_in = cars_data[cars_data[i]>upper].index
    cars_data.drop(min_in, inplace = True)
    cars_data.drop(max_in, inplace = True)
s2 = cars_data.shape
```

jupyter autos Last Checkpoint: hace un día (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

In [17]:
```
s1 = cars_data.shape
clean = cars_data[['hp', 'cylinders', 'highway_mpg', 'city_mpg', 'price']]
for i in clean.columns:
    qt1 = cars_data[i].quantile(0.25)
    qt3 = cars_data[i].quantile(0.75)
    iqr =  qt3 - qt1
    lower = qt1-(1.5*iqr)
    upper = qt3+(1.5*iqr)
    min_in = cars_data[cars_data[i]<lower].index
    max_in = cars_data[cars_data[i]>upper].index
    cars_data.drop(min_in, inplace = True)
    cars_data.drop(max_in, inplace = True)
s2 = cars_data.shape
outliers = s1[0] - s2[0]
print("Deleted outliers are : ", outliers)
```

```
Deleted outliers are :  1403
```

In [18]:
```
fig = px.box(cars_data, x = cars_data['hp'])
fig.update_traces(fillcolor = '#C9A26B')
```

jupyter autos Last Checkpoint: hace un día (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

Run Code

In [19]:
```
cars_data.describe()
```

Out[19]:

	year	hp	cylinders	doors	highway_mpg	city_mpg	popularity	price
count	9784.00000	9784.000000	9784.000000	9784.000000	9784.000000	9784.000000	9784.000000	9784.000000
mean	2010.31204	231.154334	5.424162	3.505110	26.304374	19.180192	1560.331664	29204.537612
std	7.46610	78.693348	1.395065	0.841754	5.745849	4.376362	1464.248520	15605.472026
min	1990.00000	0.000000	3.000000	2.000000	12.000000	10.000000	21.000000	2000.000000
25%	2006.00000	170.000000	4.000000	3.000000	22.000000	16.000000	549.000000	20498.000000
50%	2014.00000	220.000000	6.000000	4.000000	26.000000	18.000000	1385.000000	29089.000000
75%	2016.00000	288.000000	6.000000	4.000000	30.000000	22.000000	2009.000000	38850.000000
max	2017.00000	485.000000	8.000000	4.000000	42.000000	31.000000	5657.000000	70900.000000

In [20]:
```
for i in cars_data:
    fig = px.histogram(cars_data, x= i, color_discrete_sequence = ['#C9A26B'])
    fig.show()
```

jupyter autos Last Checkpoint: hace un día (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

Run Code

In [23]:
```
fig = go.Figure()
fig.add_trace(go.Bar(x = cars_data['transmission'].unique(), y = cars_data.groupby('transmission').mean()['hp'].sort_values(), ma
fig.update_layout(title_text = 'Car transmissions according to Hp', xaxis_title = "Car Transmission", yaxis_title = "Hp")
fig.show()
```