19120572

June 27, 2022

Họ và tên: Hồ Công Lượng

MSSV: 19120572

1 Đọc dữ liệu

[1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib as plt
import matplotlib.pyplot as plt

df = pd.read_csv("ToyotaCorolla.csv")
df
```

[1]:

	Price	Age	Kilometers	Fuel_Type	HP	Metallic	Color	Automatic	\
0	13500	23	46986	Diesel	90	1	Blue	0	
1	13750	23	72937	Diesel	90	1	Silver	0	
2	13950	24	41711	Diesel	90	1	Blue	0	
3	14950	26	48000	Diesel	90	0	Black	0	
4	13750	30	38500	Diesel	90	0	Black	0	
…	…	…	…	…	…	…	…	…	
1431	7500	69	20544	Petrol	86	1	Blue	0	
1432	10845	72	19000	Petrol	86	0	Grey	0	
1433	8500	71	17016	Petrol	86	0	Blue	0	
1434	7250	70	16916	Petrol	86	1	Grey	0	
1435	6950	76	1	Petrol	110	0	Green	0	

	CC	Doors	Quarterly_Tax	Weight
0	2000	3	210	1165
1	2000	3	210	1165
2	2000	3	210	1165
3	2000	3	210	1165
4	2000	3	210	1170
…	…	…	…	…
1431	1300	3	69	1025
1432	1300	3	69	1015
1433	1300	3	69	1015

```
1434  1300      3             69    1015
1435  1600      5             19    1114

[1436 rows x 12 columns]
```

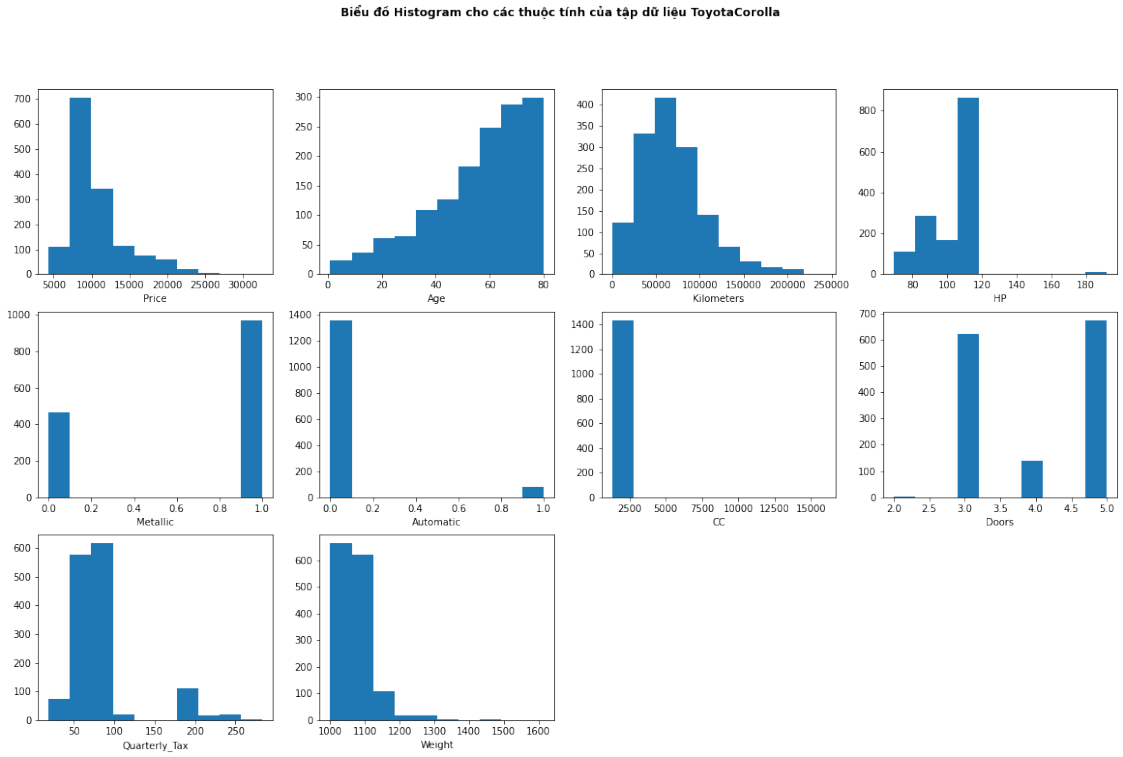
1.1 1. Hãy trực quan hóa các thông tin thống kê mô tả cho các biến.

[2]:
```
df.describe()
```

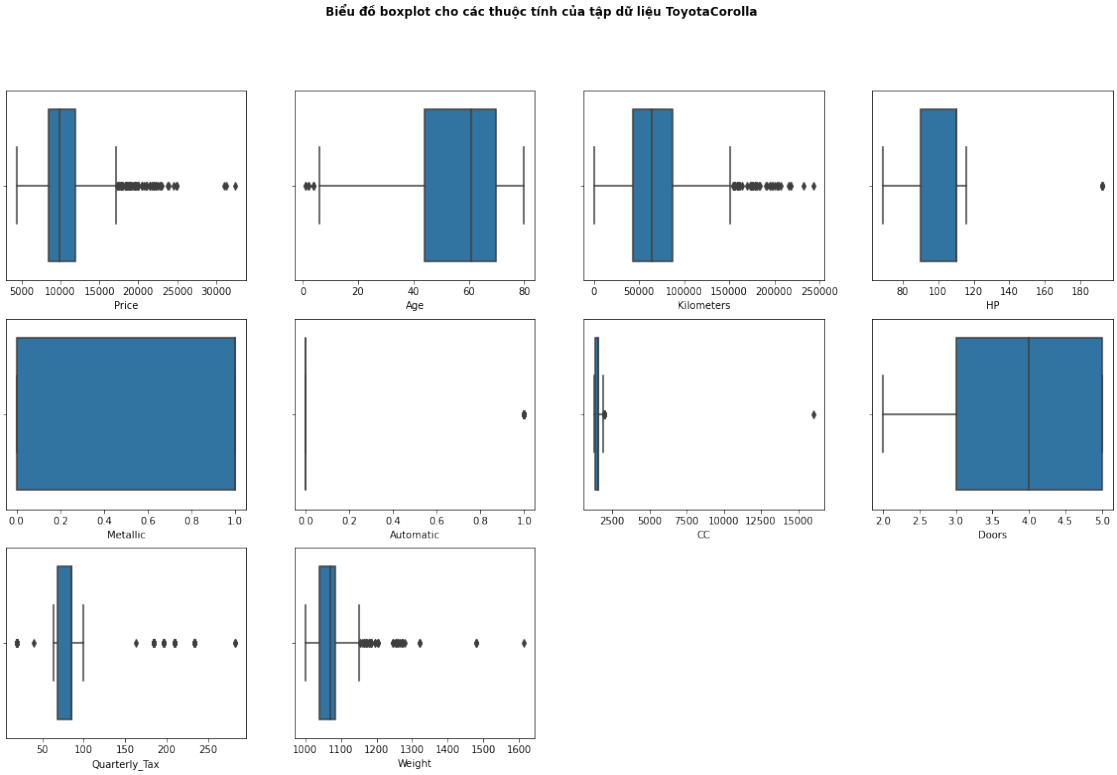
[2]:
```
              Price          Age     Kilometers           HP     Metallic  \
count   1436.000000  1436.000000    1436.000000  1436.000000  1436.000000
mean   10730.824513    55.947075   68533.259749   101.502089     0.674791
std     3626.964585    18.599988   37506.448872    14.981080     0.468616
min     4350.000000     1.000000       1.000000    69.000000     0.000000
25%     8450.000000    44.000000   43000.000000    90.000000     0.000000
50%     9900.000000    61.000000   63389.500000   110.000000     1.000000
75%    11950.000000    70.000000   87020.750000   110.000000     1.000000
max    32500.000000    80.000000  243000.000000   192.000000     1.000000

         Automatic           CC        Doors  Quarterly_Tax      Weight
count  1436.000000   1436.00000  1436.000000    1436.000000  1436.00000
mean      0.055710   1576.85585     4.033426      87.122563  1072.45961
std       0.229441    424.38677     0.952677      41.128611    52.64112
min       0.000000   1300.00000     2.000000      19.000000  1000.00000
25%       0.000000   1400.00000     3.000000      69.000000  1040.00000
50%       0.000000   1600.00000     4.000000      85.000000  1070.00000
75%       0.000000   1600.00000     5.000000      85.000000  1085.00000
max       1.000000  16000.00000     5.000000     283.000000  1615.00000
```

[3]:
```
# Vẽ biểu đồ Histogram
fig, ((ax0, ax1, ax2, ax3), (ax4, ax5, ax6, ax7), (ax8, ax9, ax10, ax11)) = plt.
 ↪subplots(nrows=3, ncols=4, figsize=(20,12))

ax0.hist(x = 'Price', data = df)
ax0.set_xlabel('Price')

ax1.hist(x = 'Age', data = df)
ax1.set_xlabel('Age')

ax2.hist(x = 'Kilometers', data = df)
ax2.set_xlabel('Kilometers')

ax3.hist(x = 'HP', data = df)
ax3.set_xlabel('HP')

ax4.hist(x = 'Metallic', data = df)
ax4.set_xlabel('Metallic')
```

```
ax5.hist(x = 'Automatic', data = df)
ax5.set_xlabel('Automatic')

ax6.hist(x = 'CC', data = df)
ax6.set_xlabel('CC')

ax7.hist(x = 'Doors', data = df)
ax7.set_xlabel('Doors')

#ax8 = fig.add_subplot(149)
ax8.hist(x = 'Quarterly_Tax', data = df)
ax8.set_xlabel('Quarterly_Tax')

ax9.hist(x = 'Weight', data = df)
ax9.set_xlabel('Weight')

fig.delaxes(ax10)
fig.delaxes(ax11)
fig.suptitle('Biểu đồ Histogram cho các thuộc tính của tập dữ liệu␣
 ↪ToyotaCorolla', fontweight = 'bold')
plt.show()
```



[4]:
```
# Vẽ biểu đồ boxplot
fig, ((ax0, ax1, ax2, ax3), (ax4, ax5, ax6, ax7), (ax8, ax9, ax10, ax11)) = plt.
 ↪subplots(nrows=3, ncols=4, figsize=(20,12))

sns.boxplot(ax = ax0, x = 'Price', data = df)

sns.boxplot(ax = ax1, x = 'Age', data = df)

sns.boxplot(ax = ax2, x = 'Kilometers', data = df)

sns.boxplot(ax = ax3, x = 'HP', data = df)

sns.boxplot(ax = ax4, x = 'Metallic', data = df)

sns.boxplot(ax = ax5, x = 'Automatic', data = df)

sns.boxplot(ax = ax6, x = 'CC', data = df)

sns.boxplot(ax = ax7, x = 'Doors', data = df)

sns.boxplot(ax = ax8, x = 'Quarterly_Tax', data = df)

sns.boxplot(ax = ax9, x = 'Weight', data = df)

fig.delaxes(ax10)
fig.delaxes(ax11)
fig.suptitle('Biểu đồ boxplot cho các thuộc tính của tập dữ liệu␣
 ↪ToyotaCorolla', fontweight = 'bold')
plt.show()
```

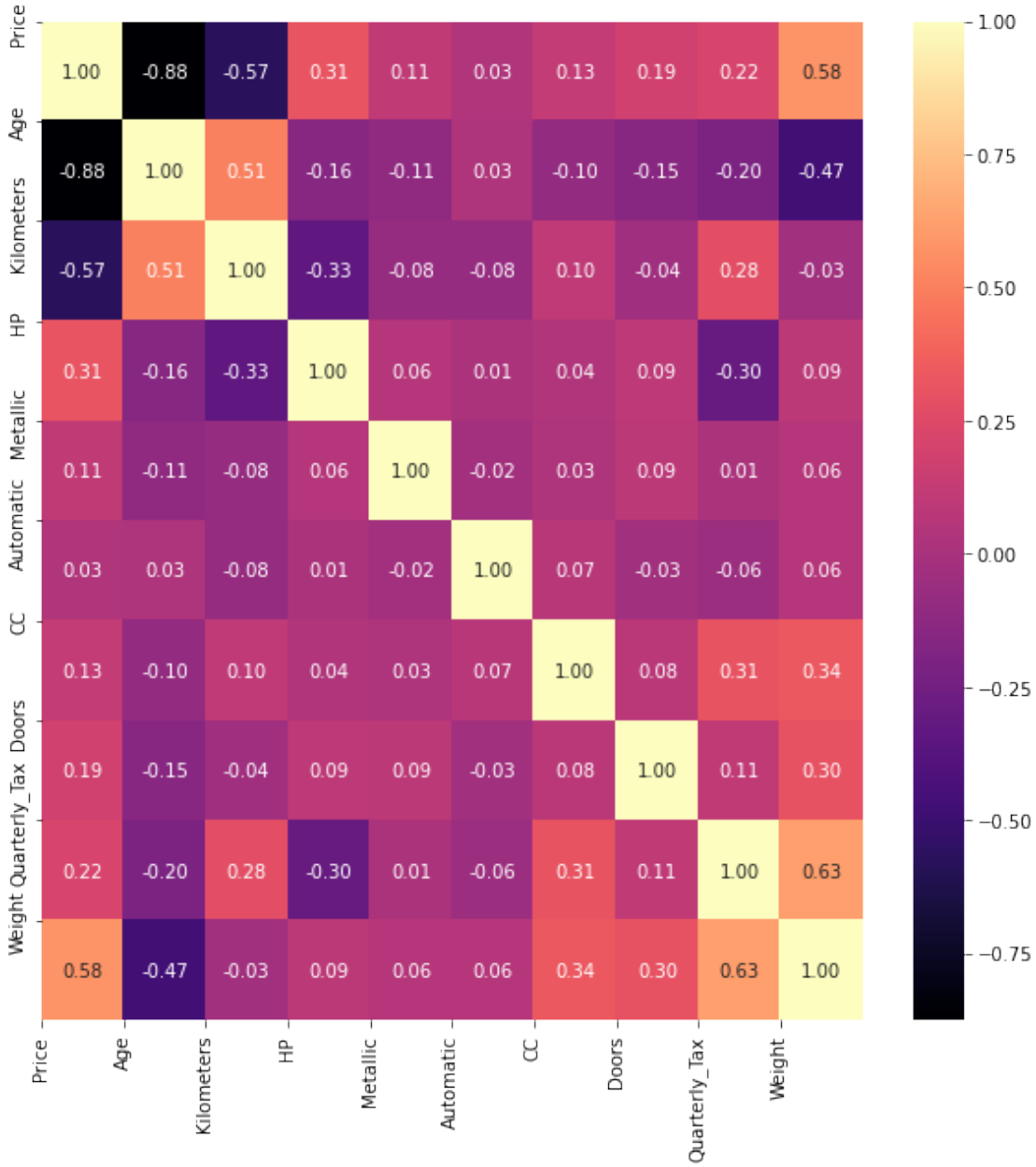


1.1.1 Nhận xét:

- Tất cả các thuộc tính của tập dữ liệu đều không phải phân phối chuẩn.
- Price, Age, Kilometers, HP, Quarterly_Tax và Weight có tồn tại các outliers.
- Các thuộc tính phân bố không đồng đều. Ví dụ Price, Quarterly_Tax, Weight bị lệch phải, còn Age, HP bị lệch trái.

1.2 2. Tìm và trực quan mối quan hệ tương quan giữa các cặp biến (nếu có)

[5]:
```
corr = df.corr()
fig, ax = plt.subplots(figsize=(10, 10))

sns.heatmap(corr, cmap='magma', annot=True, fmt=".2f")

plt.xticks(range(len(corr.columns)), corr.columns);
plt.yticks(range(len(corr.columns)), corr.columns)

plt.show()
```

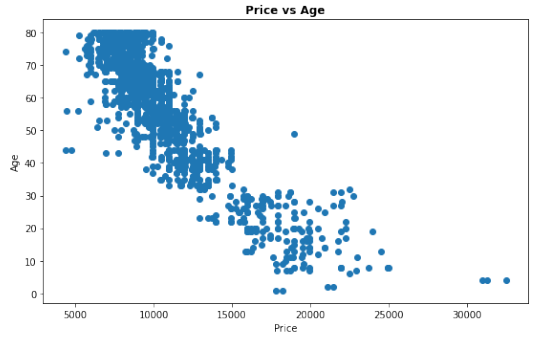


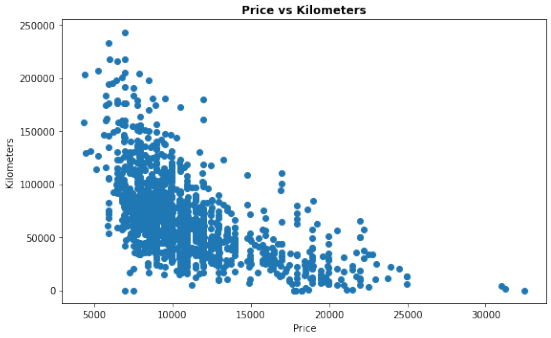
Nhìn vào biểu đồ, ta thấy được một số cặp thuộc tính có độ tương quan khá cao, đó là:

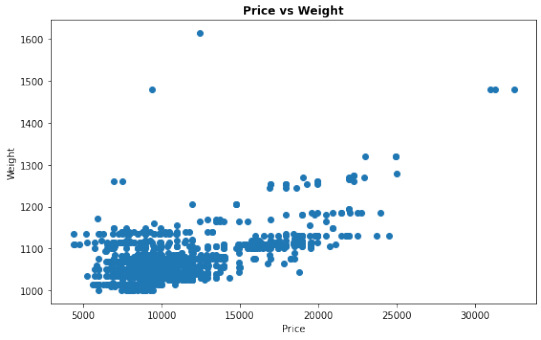
- Price vs Age
- Price vs Kilometers
- Price vs Weight

[6]:
```
fig, ((ax0, ax1), (ax2, ax3)) = plt.subplots(nrows=2, ncols=2, figsize=(20,12))

# Price vs Age
x1 = df['Price']
```

```
y1 = df['Age']
ax0.scatter(x1, y1)
ax0.set_xlabel("Price")
ax0.set_ylabel("Age")
ax0.set_title("Price vs Age", fontweight = 'bold')

# Price vs Kilometers
x2 = df['Price']
y2 = df['Kilometers']
ax1.scatter(x2, y2)
ax1.set_xlabel("Price")
ax1.set_ylabel("Kilometers")
ax1.set_title("Price vs Kilometers", fontweight = 'bold')

# Price vs Weight
x3 = df['Price']
y3 = df['Weight']
ax2.scatter(x3, y3)
ax2.set_xlabel("Price")
ax2.set_ylabel("Weight")
ax2.set_title("Price vs Weight", fontweight = 'bold')

fig.delaxes(ax3)
plt.show()
```







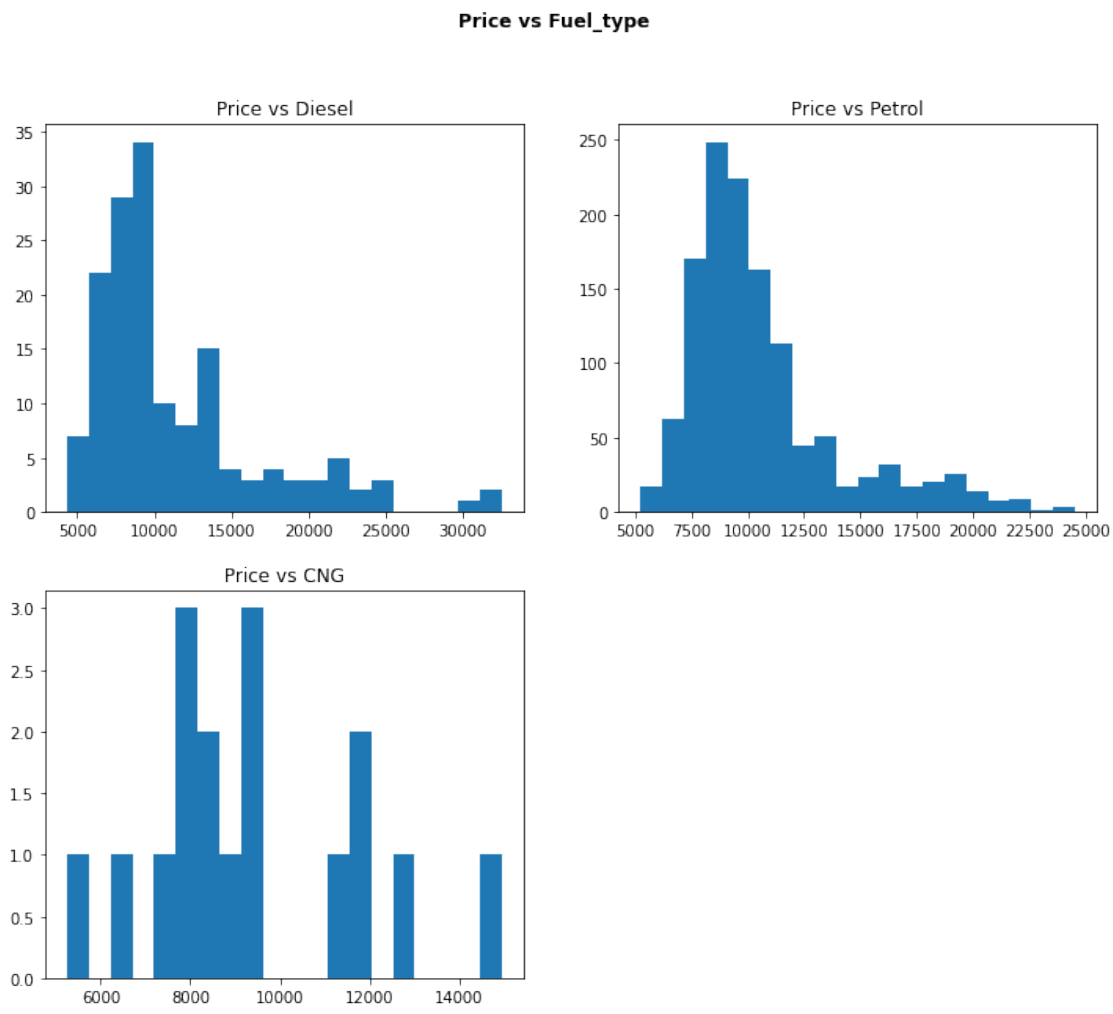
Nhận xét:

- (Price vs Age) và (Price vs Kilometers) có mối quan hệ tương quan âm với nhau.
- (Price vs Weight) có mối quan hệ tương quan dương với nhau. (hay giá trị của xe tăng theo trọng lượng xe)
- Giá trị tương quan giữa Price vs Age khá cao, điều này cho thấy Price với Age có mối quan hệ khá chặt chẽ với nhau. (có nghĩa là xe càng cũ thì giá trị của xe càng nhỏ)

1.3 3. Hãy trực quan hóa biểu đồ histogram cho Price theo từng biến biến

- Fuel_type
- Color

[7]:
```
# PRICE VS FUEL_TYPE

Fuel_type = df['Fuel_Type'].unique()

fig, ((ax0, ax1), (ax2, ax3)) = plt.subplots(nrows=2, ncols=2, figsize=(12,10))

ax0.hist(df[df['Fuel_Type'] == Fuel_type[0]]['Price'], density=0, bins = 20)
ax0.set_title("Price vs " + str(Fuel_type[0]))

ax1.hist(df[df['Fuel_Type'] == Fuel_type[1]]['Price'], density=0, bins = 20)
ax1.set_title("Price vs " + str(Fuel_type[1]))

ax2.hist(df[df['Fuel_Type'] == Fuel_type[2]]['Price'], density=0, bins = 20)
ax2.set_title("Price vs " + str(Fuel_type[2]))

fig.suptitle('Price vs Fuel_type',fontweight ="bold")
fig.delaxes(ax3)
plt.show()
```

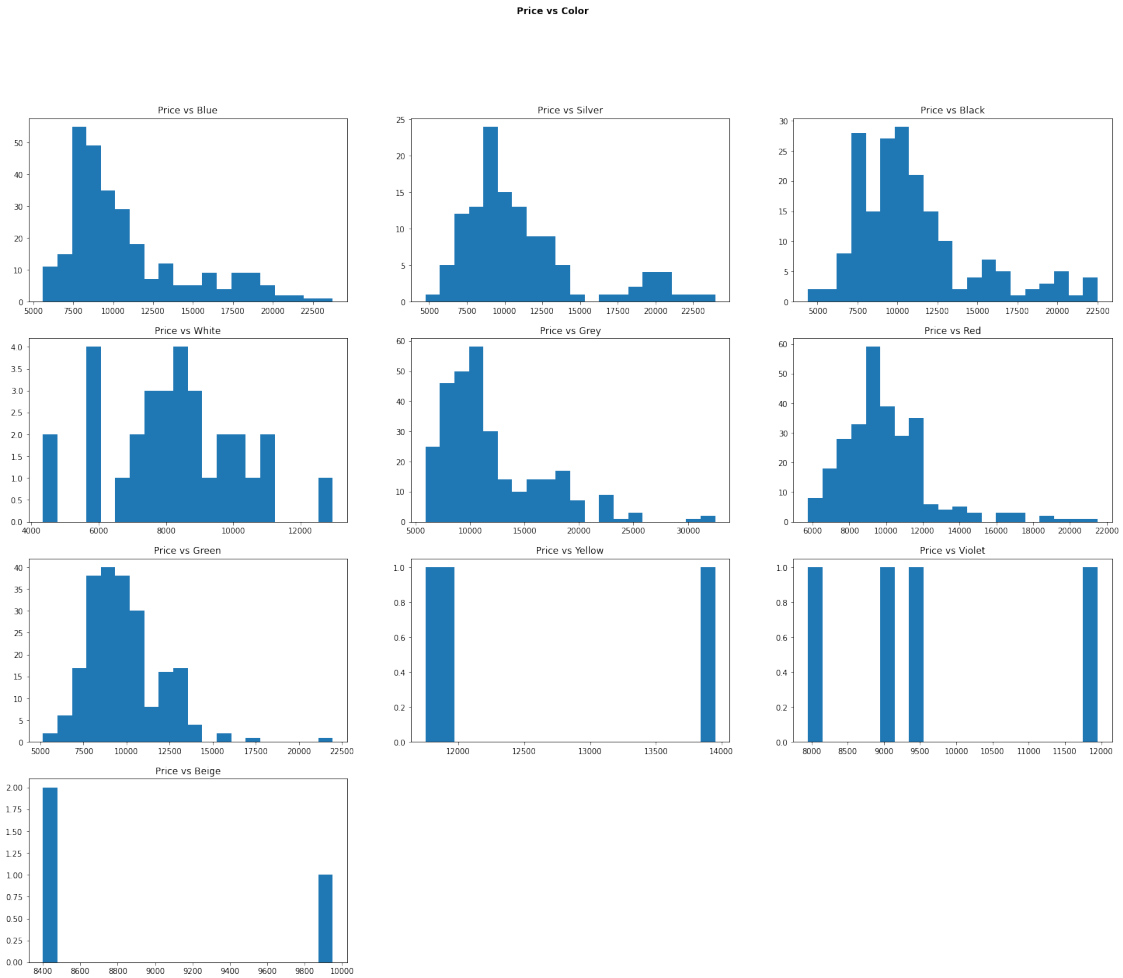


Nhận xét:

- Các biểu đồ có xu hướng lệch phải.
- Xe sử dụng Petrol chiếm số lượng áp đảo so với Diesel và CNG.

[8]:
```
# PRICE VS COLOR

Color = df['Color'].unique()

nrows=4
ncols=3
temp = 0

fig, ax = plt.subplots(nrows=nrows, ncols=ncols, figsize=(25,20))

for i in range(nrows):
  for j in range(ncols):
```

```
    if i == nrows - 1 and j == 1:
      break
    ax[i][j].hist(df[df['Color'] == Color[temp]]['Price'], density=0, bins = 20)
    ax[i][j].set_title("Price vs " + str(Color[temp]))

    temp = temp + 1

fig.suptitle('Price vs Color', fontweight ="bold")
fig.delaxes(ax[3][1])
fig.delaxes(ax[3][2])
plt.show()
```



Nhận xét:

- Một số biểu đồ có xu hướng lệch phải như Price vs Blue, Price vs Silver,..
- Số lượng xe được thiết kế với màu Yellow, Violet, Beige rất ít.

1.4 4. Hãy đưa ra mô hình dự báo về giá xe Price (có thể sử dụng mô hình hồi quy logistic hoặc mô hình học máy bất kỳ).

[9]:
```
# Thực hiện xây dựng mô hình dự báo về giá của Price với các thuộc tính Age,␣
 ↪Kilometers, Weight.
# Sử dụng mô hình Ordinary least squares (OLS)

import statsmodels.api as sm

x = df[['Age', 'Kilometers', 'Weight']]
x = np.reshape(x, (-1, 3))
y = np.array(df["Price"])
y = np.reshape(y, (-1, 1))

X = sm.add_constant(x)
model = sm.OLS(y, X)

result = model.fit()
print(result.summary())
```

```
/usr/local/lib/python3.7/dist-packages/statsmodels/tools/_testing.py:19:
FutureWarning: pandas.util.testing is deprecated. Use the functions in the
public API at pandas.testing instead.
  import pandas.util.testing as tm
```

OLS Regression Results

Dep. Variable:	y	R-squared:	0.848
Model:	OLS	Adj. R-squared:	0.848
Method:	Least Squares	F-statistic:	2665.
Date:	Mon, 27 Jun 2022	Prob (F-statistic):	0.00
Time:	08:29:21	Log-Likelihood:	-12454.
No. Observations:	1436	AIC:	2.492e+04
Df Residuals:	1432	BIC:	2.494e+04
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-1880.3356	962.718	-1.953	0.051	-3768.825	8.153
Age	-120.2212	2.742	-43.841	0.000	-125.600	-114.842
Kilometers	-0.0242	0.001	-20.142	0.000	-0.027	-0.022
Weight	19.5760	0.836	23.409	0.000	17.936	21.216

Omnibus:	221.061	Durbin-Watson:	1.523
Prob(Omnibus):	0.000	Jarque-Bera (JB):	2197.082
Skew:	-0.373	Prob(JB):	0.00
Kurtosis:	9.013	Cond. No.	2.01e+06

```
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 2.01e+06. This might indicate that there are
strong multicollinearity or other numerical problems.

/usr/local/lib/python3.7/dist-packages/statsmodels/tsa/tsatools.py:117:
FutureWarning: In a future version of pandas all arguments of concat except for
the argument 'objs' will be keyword-only
  x = pd.concat(x[::order], 1)
```

[10]:
```
print("Các thông số của mô hình: \n", result.params)
```

```
Các thông số của mô hình:
 const        -1880.335564
Age           -120.221174
Kilometers      -0.024183
Weight          19.576043
dtype: float64
```

Vậy, mô hình dự báo về giá xe (Price) bằng các thuộc tính Age, Kilometes, Weight là:

1.4.1 > *Price = -1880.335564 - 120.221174.Age - 0.024183.Kilometers + 19.576043.Weight*

1.4.2 ==> Mô hình cho ra giá trị R-squared = 0.848. Đây là giá trị khá cao nên ta có thể thấy đây là mô hình khá tốt để dự đoán giá xe.