Exploratory Data Analysis

SW

2025-01-26

Exploratory Data Analysis

Objectives

After completing this lab you will be able to:

- Explore features or characteristics to predict price of car
- Analyze patterns and run descriptive statistical analysis
- Group data based on identified parameters and create pivot tables
- Identify the effect of independent attributes on price of cars

Table of Contents

Import Data from Module

Analyzing Individual Feature Patterns using Visualization

Descriptive Statistical Analysis

Basics of Grouping

Correlation and Causation

Import Data from Module 2

Setup Import libraries:

```
import pandas as pd
import numpy as np
```

Download the dataset and store it in dataframe `df`:

```
filepath='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/\
IBMDeveloperSkillsNetwork-DA0101EN-SkillsNetwork/labs/Data%20files/automobileEDA.csv'
df = pd.read_csv(filepath, header=0)
```

View the first 5 values of the dataframe using `datafame.head()`:

```
df.head()
```

```
##    symboling  normalized-losses         make  ... horsepower-binned diesel gas
## 0          3                122  alfa-romero  ...             Medium      0   1
## 1          3                122  alfa-romero  ...             Medium      0   1
## 2          1                122  alfa-romero  ...             Medium      0   1
## 3          2                164         audi  ...             Medium      0   1
## 4          2                164         audi  ...             Medium      0   1
##
## [5 rows x 29 columns]
```

Analyzing Individual Feature Patterns Using Visualization

Import visualization packages "Matplotlib" and "Seaborn". Don't forget about "%matplotlib inline" to plot in a Jupyter notebook.

```
import matplotlib.pyplot as plt
import seaborn as sns
```

```
# list the data types for each column
print(df.dtypes)
```

```
## symboling              int64
## normalized-losses      int64
## make                  object
## aspiration            object
## num-of-doors          object
## body-style            object
## drive-wheels          object
## engine-location       object
## wheel-base           float64
## length               float64
## width                float64
## height               float64
## curb-weight            int64
## engine-type           object
## num-of-cylinders      object
## engine-size            int64
## fuel-system           object
## bore                 float64
## stroke               float64
## compression-ratio    float64
## horsepower           float64
## peak-rpm             float64
## city-mpg               int64
## highway-mpg            int64
## price                float64
## city-L/100km         float64
## horsepower-binned     object
## diesel                 int64
## gas                    int64
## dtype: object
```

```
numeric_df = df.select_dtypes(include=['float64', 'int64'])
numeric_df.corr()
```

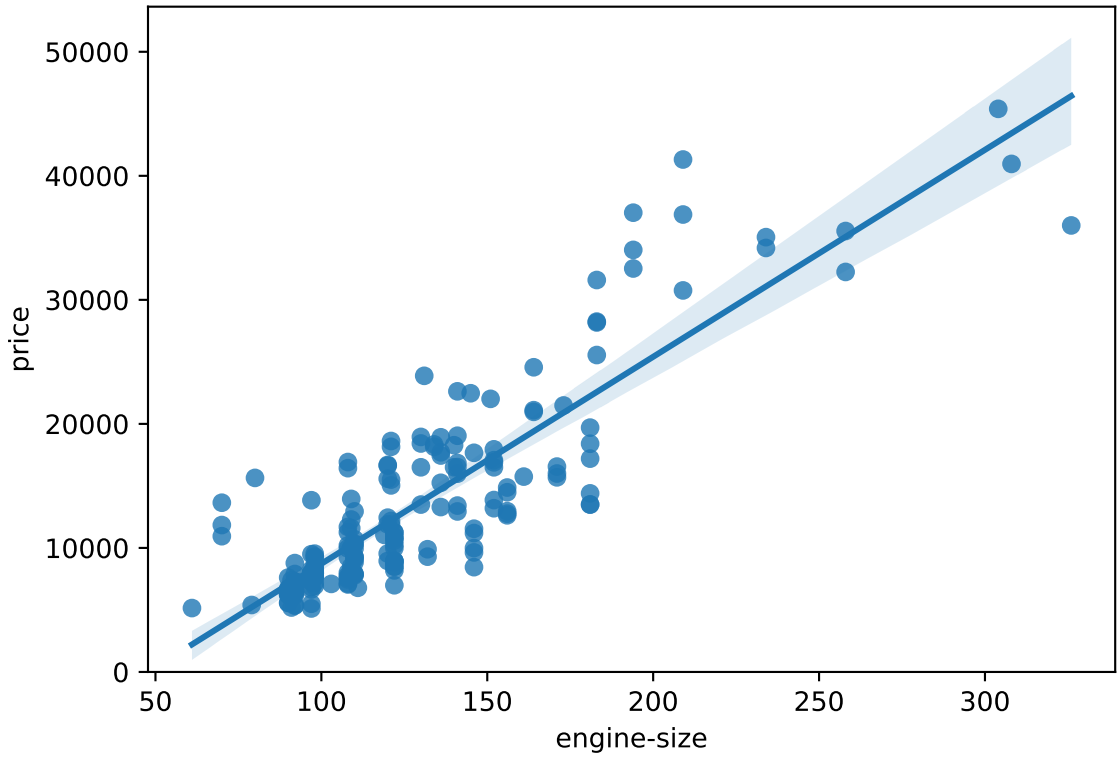
```
##                    symboling  normalized-losses  ...    diesel       gas
## symboling           1.000000           0.466264  ... -0.196735  0.196735
## normalized-losses   0.466264           1.000000  ... -0.101546  0.101546
## wheel-base         -0.535987          -0.056661  ...  0.307237 -0.307237
## length             -0.365404           0.019424  ...  0.211187 -0.211187
## width              -0.242423           0.086802  ...  0.244356 -0.244356
## height             -0.550160          -0.373737  ...  0.281578 -0.281578
## curb-weight        -0.233118           0.099404  ...  0.221046 -0.221046
## engine-size        -0.110581           0.112360  ...  0.070779 -0.070779
## bore               -0.140019          -0.029862  ...  0.054458 -0.054458
## stroke             -0.008245           0.055563  ...  0.241303 -0.241303
## compression-ratio  -0.182196          -0.114713  ...  0.985231 -0.985231
## horsepower          0.075819           0.217299  ... -0.169053  0.169053
## peak-rpm            0.279740           0.239543  ... -0.475812  0.475812
## city-mpg           -0.035527          -0.225016  ...  0.265676 -0.265676
## highway-mpg         0.036233          -0.181877  ...  0.198690 -0.198690
## price              -0.082391           0.133999  ...  0.110326 -0.110326
## city-L/100km        0.066171           0.238567  ... -0.241282  0.241282
## diesel             -0.196735          -0.101546  ...  1.000000 -1.000000
## gas                 0.196735           0.101546  ... -1.000000  1.000000
##
## [19 rows x 19 columns]
```

```
df[['bore', 'stroke', 'compression-ratio', 'horsepower']].corr()
```

```
##                      bore    stroke  compression-ratio  horsepower
## bore             1.000000 -0.055390           0.001263    0.566936
## stroke          -0.055390  1.000000           0.187923    0.098462
## compression-ratio  0.001263  0.187923           1.000000   -0.214514
## horsepower       0.566936  0.098462          -0.214514    1.000000
```

```
# Engine size as potential predictor variable of price
sns.regplot(x="engine-size", y="price", data=df)
plt.ylim(0,)
```

```
## (0.0, 53637.15290863124)
```



Reference

How to Disable Warnings in Jupyter Notebook