TASK

Exploratory Data Analysis on the Automobile Data Set

Visit our website

Introduction

Summary of the data set

DATA CLEANING

The following techniques were carried out during data cleaning

Looking into the data

- In order to get the idea and the nature of the dataset, the dataset was inspected in order to identify the data types for each column and identify the field that may not be of high value to the analysis of our interest

```
In [5]: # display summary of the data
        df_data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 205 entries, 0 to 204
Data columns (total 26 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   symboling          205 non-null    int64
 1   normalized-losses  164 non-null    object
 2   make               205 non-null    object
 3   fuel-type          205 non-null    object
 4   aspiration         205 non-null    object
 5   num-of-doors       203 non-null    object
 6   body-style         205 non-null    object
 7   drive-wheels       205 non-null    object
 8   engine-location    205 non-null    object
 9   wheel-base         205 non-null    float64
 10  length             205 non-null    float64
 11  width              205 non-null    float64
 12  height             205 non-null    float64
 13  curb-weight        205 non-null    int64
 14  engine-type        205 non-null    object
 15  num-of-cylinders   205 non-null    object
 16  engine-size        205 non-null    int64
 17  fuel-system        205 non-null    object
 18  bore               201 non-null    object
 19  stroke             201 non-null    object
 20  compression-ratio  205 non-null    float64
 21  horsepower         203 non-null    object
 22  peak-rpm           203 non-null    object
 23  city-mpg           205 non-null    int64
 24  highway-mpg        205 non-null    int64
 25  price              201 non-null    object
dtypes: float64(5), int64(5), object(16)
memory usage: 41.8+ KB
```

Replacing the question marks with NaN

- In order to get the best out of our analysis, non_categorical feature within the value column of our dataset are identified, and to ensure that the appropriates techniques is implemented on the non_categorical observation

In [3]:
```python
# The data contains question marks. We can replace it with NAN
df_data = df.replace('?',np.NAN)
df_data.isnull().sum()
```

Out[3]:
```
symboling             0
normalized-losses    41
make                  0
fuel-type             0
aspiration            0
num-of-doors          2
body-style            0
drive-wheels          0
engine-location       0
wheel-base            0
length                0
width                 0
height                0
curb-weight           0
engine-type           0
num-of-cylinders      0
engine-size           0
fuel-system           0
bore                  4
stroke                4
compression-ratio     0
horsepower            2
peak-rpm              2
city-mpg              0
highway-mpg           0
price                 4
dtype: int64
```

Data reduction

- Here we dropped features if they do not sum up value to our analysis

In [15]:
```python
# code here
delete_col_list = ['symboling', 'normalized-losses']
automobile_df = automobile_df.drop(delete_col_list, axis=1)
automobile_df.head()
```

Out[15]:

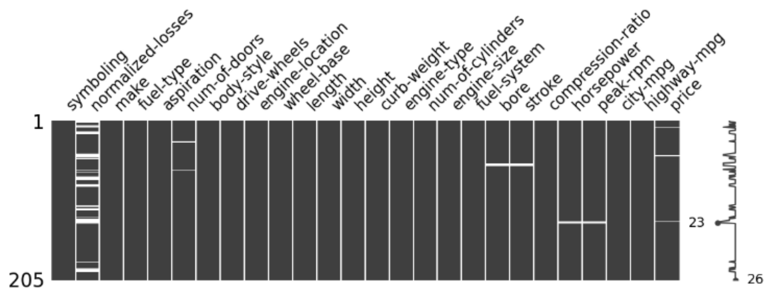
	make	fuel-type	aspiration	num-of-doors	body-style	drive-wheels	engine-location	wheel-base	length	width	...	engine-size	fuel-system	bore	stroke	compression-ratio	horsepower	peak-rpm	ci m
0	alfa-romero	gas	std	two	convertible	rwd	front	88.6	168.8	64.1	...	130	mpfi	3.47	2.68	9.0	111	5000	
1	alfa-romero	gas	std	two	convertible	rwd	front	88.6	168.8	64.1	...	130	mpfi	3.47	2.68	9.0	111	5000	
2	alfa-romero	gas	std	two	hatchback	rwd	front	94.5	171.2	65.5	...	152	mpfi	2.68	3.47	9.0	154	5000	
3	audi	gas	std	four	sedan	fwd	front	99.8	176.6	66.2	...	109	mpfi	3.19	3.40	10.0	102	5500	
4	audi	gas	std	four	sedan	4wd	front	99.4	176.6	66.4	...	136	mpfi	3.19	3.40	8.0	115	5500	

5 rows × 24 columns

MISSING DATA

- There are missing observation within the dataset
- In the case of the automobile dataset, there are incognisant number of missing values, there it is the best to remove the observations with missing values

```
In [9]: # Visualise missing data
        missingno.matrix(df_data, figsize = (13,3))
Out[9]: <AxesSubplot:>
```



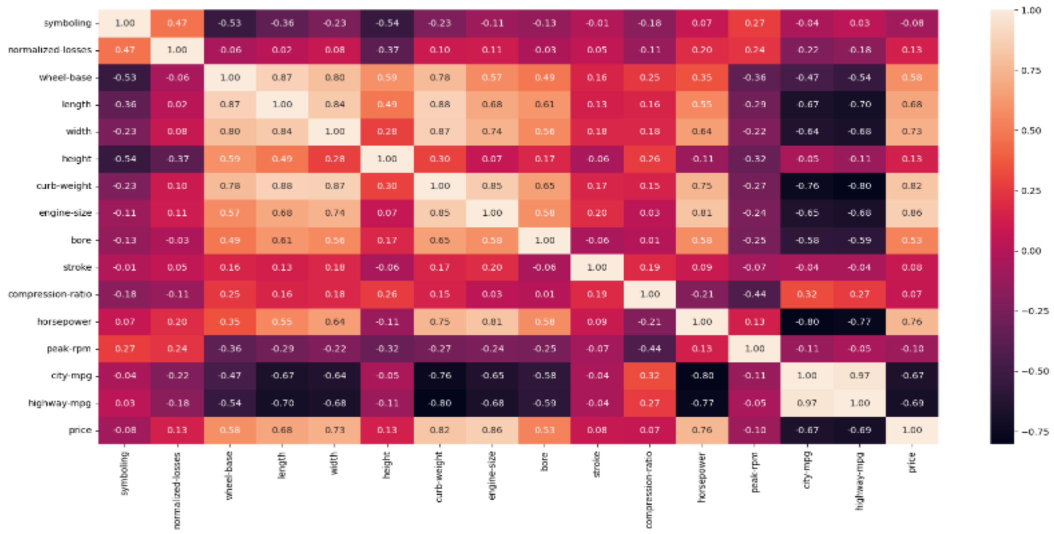

```
In [10]: # Number of missing data points per column
         missing_values_count = df_data.isnull().sum()
         missing_values_count
Out[10]: symboling             0
         normalized-losses    41
         make                  0
         fuel-type             0
         aspiration            0
         num-of-doors          2
         body-style            0
         drive-wheels          0
         engine-location       0
         wheel-base            0
         length                0
         width                 0
         height                0
         curb-weight           0
         engine-type           0
         num-of-cylinders      0
         engine-size           0
         fuel-system           0
         bore                  4
         stroke                4
         compression-ratio     0
         horsepower            2
         peak-rpm              2
         city-mpg              0
         highway-mpg           0
         price                 4
         dtype: int64
```

DATA STORIES AND VISUALISATIONS

Correlation Analysis

The heatmap shows that some variables have a high correlation with each other, such as engine size and horsepower, curb weight and length, highway mpg and city mpg, etc. These variables tend to move together in the same or opposite direction, indicating a linear relationship. The heatmap also shows that some variables have a low or no correlation with each other, such as symboling and bore, peak rpm and compression ratio, etc. These variables do not show any clear pattern of association, indicating a lack of relationship

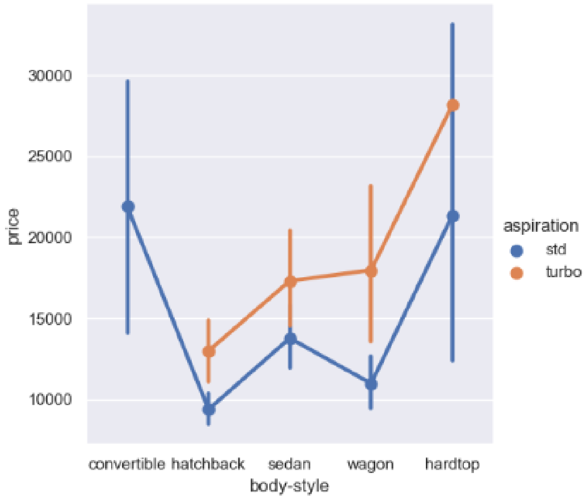
```
In [21]: corr = df.corr()
         plt.figure(figsize=(20,9))
         a = sns.heatmap(corr, annot=True, fmt='.2f')
```



The catplot

The categorical plot shows that the convertible and hardtop cars have the highest average prices, while the hatchback cars have the lowest. The sedan and wagon cars have similar average prices, but the wagon cars have a wider range of prices. The plot also shows that the turbo cars have higher average prices than the standard cars for all body styles, except for the wagon cars, where the difference is not significant. The plot also shows that the effect of aspiration is more pronounced for the convertible and hardtop cars than for the other body styles.

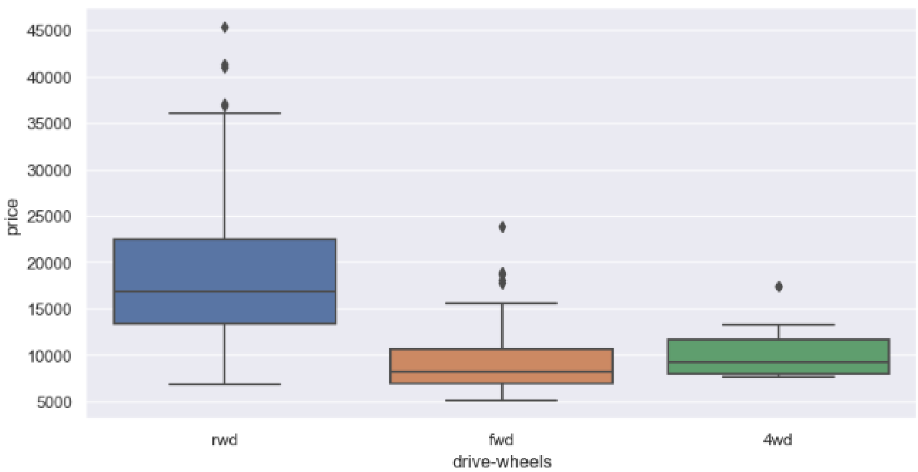
```
In [48]: sns.catplot(data=df, x="body-style", y="price", hue="aspiration" ,kind="point")
Out[48]: <seaborn.axisgrid.FacetGrid at 0x1712be0a580>
```



Box plot

The boxplot shows the distribution of prices for different types of drive wheels. We can see that the rear wheel drive cars have the highest median and range of prices, while the front wheel drive cars have the lowest. The four wheel drive cars have a slightly higher median price than the front wheel drive cars, but a lower range. However, we should note that there are only a few observations for the four wheel drive cars in our dataset, so this result may not be very reliable.

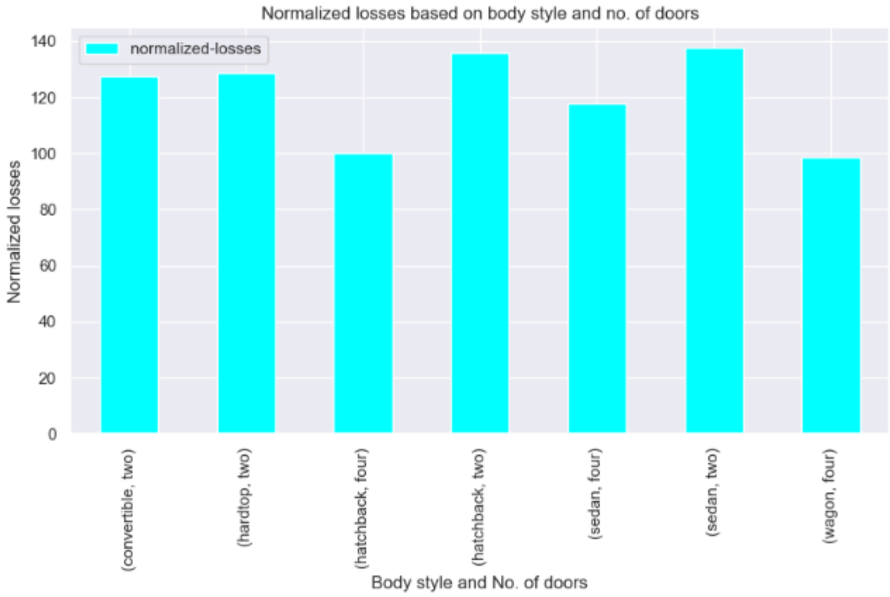
```
In [49]: plt.rcParams['figure.figsize']=(10,5)
         ax = sns.boxplot(x="drive-wheels", y="price", data=df)
```



Normalized losses based on body style and no. of doors

The normalized loss is a measure of how much a car costs to insure, based on various factors such as body style and number of doors. The boxplot shows how the normalized loss varies for different combinations of these factors. We can see that the two door cars tend to have higher normalized losses than the four door cars, regardless of the body style. This suggests that the two door cars are more risky or expensive to insure than the four door cars.

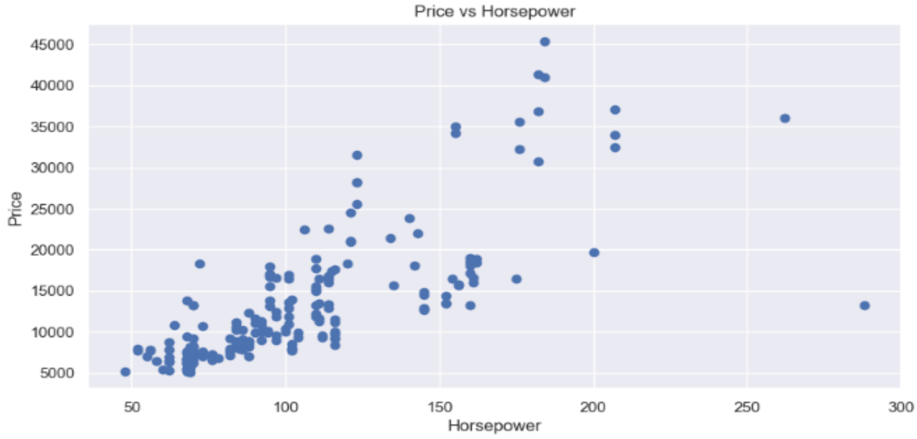
In [51]:
```
# Normalized losses based on body style and no. of doors
pd.pivot_table(df,index=['body-style','num-of-doors'], values='normalized-losses').plot(kind='bar',color='cyan')
plt.title("Normalized losses based on body style and no. of doors")
plt.ylabel('Normalized losses')
plt.xlabel('Body style and No. of doors');
```



Price vs Horsepower

The output the code is a graphical representation of the relationship between price and horsepower for the cars in the dataset. The scatter plot shows that there is a positive correlation between price and horsepower, meaning that as the horsepower increases, the price also tends to increase. However, the correlation is not very strong, as there are some outliers and variations in the data. For example, there are some cars with low horsepower but high price, and some cars with high horsepower but low price. The scatter plot also shows that most of the cars have a horsepower between 50 and 150, and a price between 5000 and 20000.

In [53]:
```
plt.scatter(df['horsepower'], df['price'])
plt.xlabel('Horsepower')
plt.ylabel('Price')
plt.title('Price vs Horsepower')
plt.show()
```



THIS REPORT WAS WRITTEN BY : Ndatadzeyi B Chiota