LAB # 10

Fall 2021

Data Analytics Lab

Submitted by: **Shah Raza**

Registration No.: **18PWCSE1658**

Section: **B**

“On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work.”

Student Signature: ______________

Submitted to:

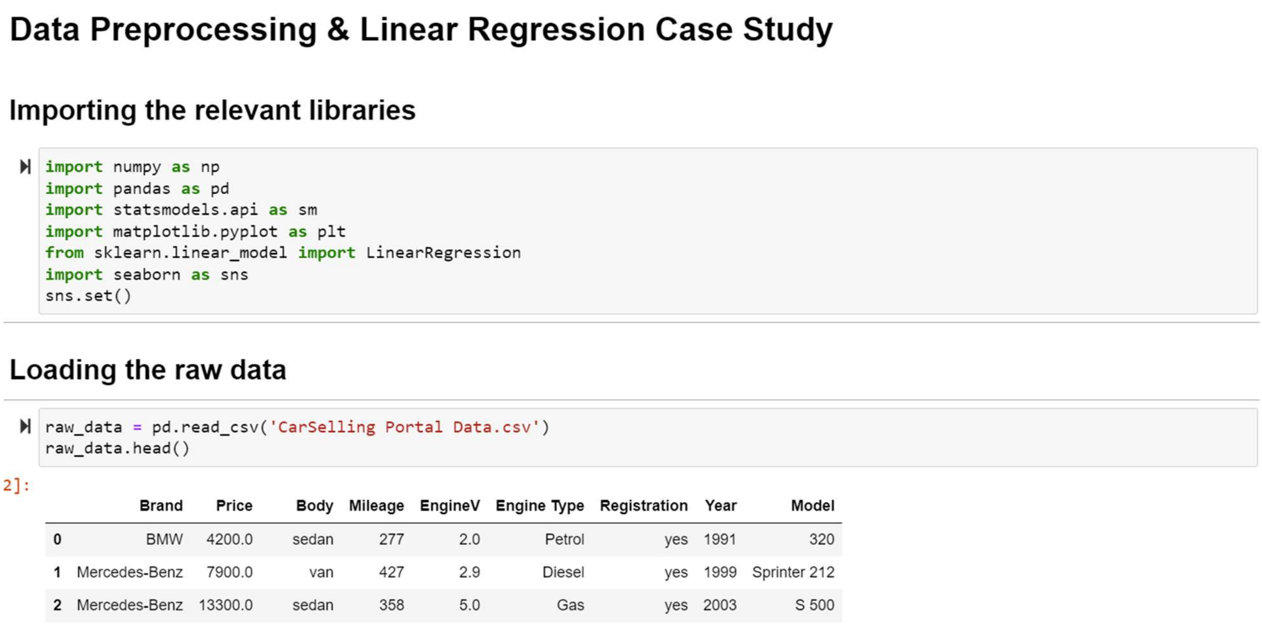
Engr. Mian Ibad Ali Shah

February 24, 2022

Department of Computer Systems Engineering

University of Engineering and Technology, Peshawar

TASK:



Data Preprocessing & Linear Regression Case Study

Importing the relevant libraries

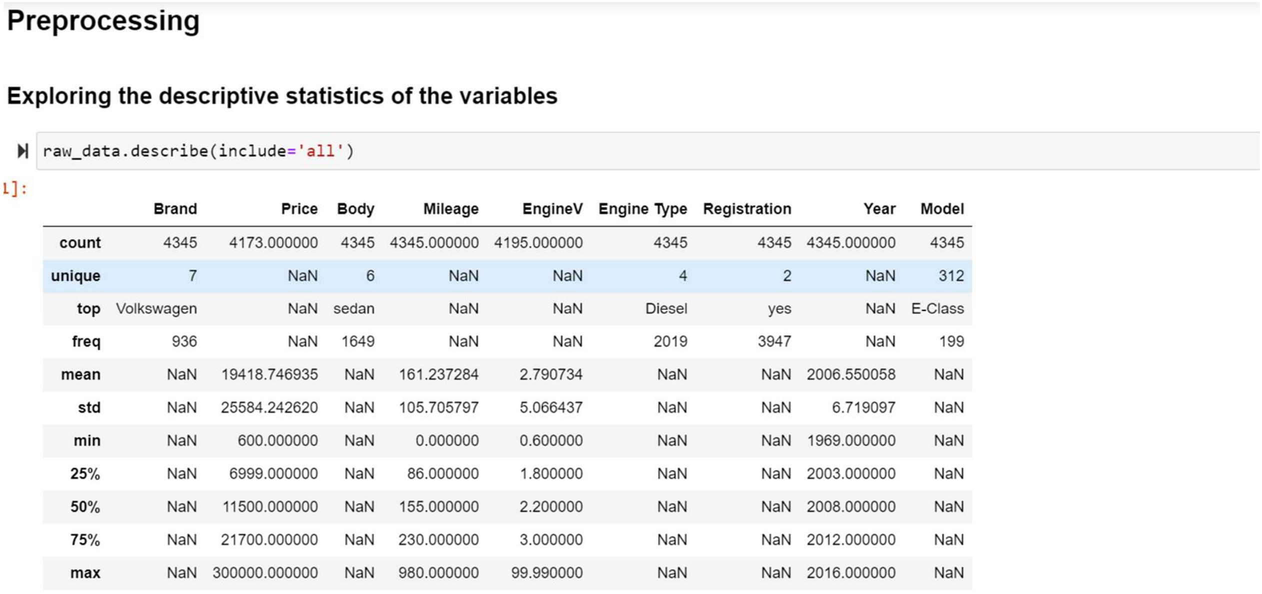
```
import numpy as np
import pandas as pd
import statsmodels.api as sm
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
import seaborn as sns
sns.set()
```

Loading the raw data

```
raw_data = pd.read_csv('CarSelling Portal Data.csv')
raw_data.head()
```

2]:

	Brand	Price	Body	Mileage	EngineV	Engine Type	Registration	Year	Model
0	BMW	4200.0	sedan	277	2.0	Petrol	yes	1991	320
1	Mercedes-Benz	7900.0	van	427	2.9	Diesel	yes	1999	Sprinter 212
2	Mercedes-Benz	13300.0	sedan	358	5.0	Gas	yes	2003	S 500



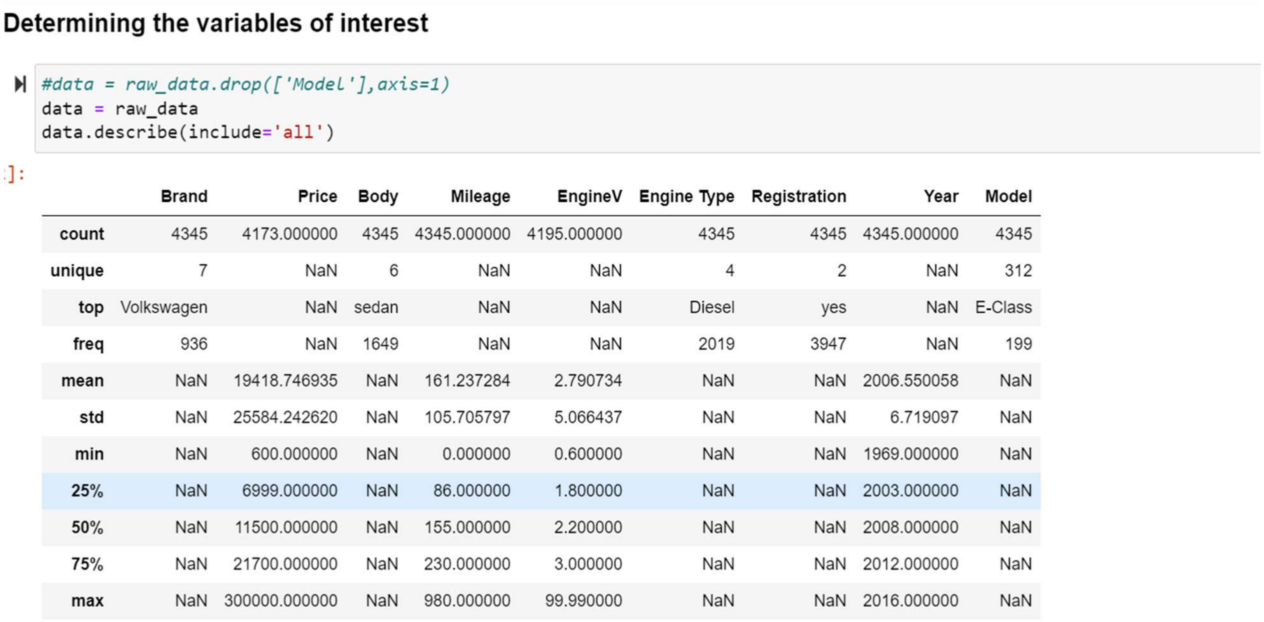
Preprocessing

Exploring the descriptive statistics of the variables

```
raw_data.describe(include='all')
```

1]:

	Brand	Price	Body	Mileage	EngineV	Engine Type	Registration	Year	Model
count	4345	4173.000000	4345	4345.000000	4195.000000	4345	4345	4345.000000	4345
unique	7	NaN	6	NaN	NaN	4	2	NaN	312
top	Volkswagen	NaN	sedan	NaN	NaN	Diesel	yes	NaN	E-Class
freq	936	NaN	1649	NaN	NaN	2019	3947	NaN	199
mean	NaN	19418.746935	NaN	161.237284	2.790734	NaN	NaN	2006.550058	NaN
std	NaN	25584.242620	NaN	105.705797	5.066437	NaN	NaN	6.719097	NaN
min	NaN	600.000000	NaN	0.000000	0.600000	NaN	NaN	1969.000000	NaN
25%	NaN	6999.000000	NaN	86.000000	1.800000	NaN	NaN	2003.000000	NaN
50%	NaN	11500.000000	NaN	155.000000	2.200000	NaN	NaN	2008.000000	NaN
75%	NaN	21700.000000	NaN	230.000000	3.000000	NaN	NaN	2012.000000	NaN
max	NaN	300000.000000	NaN	980.000000	99.990000	NaN	NaN	2016.000000	NaN

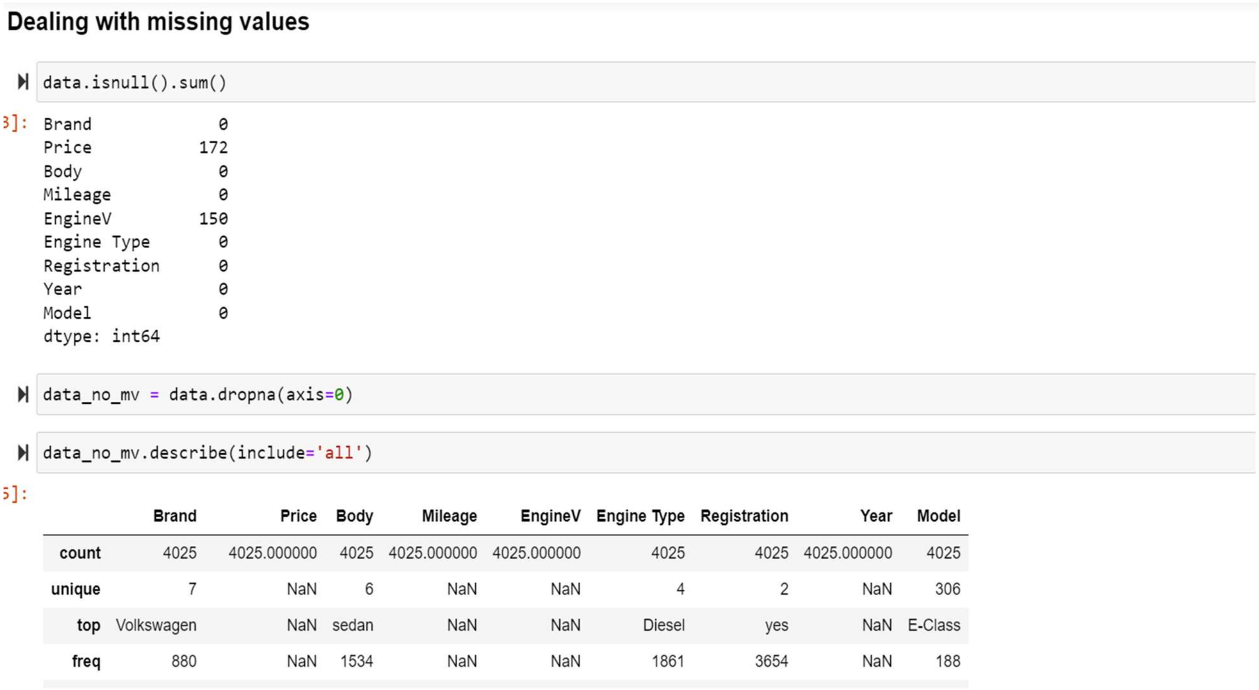


Determining the variables of interest

```
#data = raw_data.drop(['Model'],axis=1)
data = raw_data
data.describe(include='all')
```

]:

	Brand	Price	Body	Mileage	EngineV	Engine Type	Registration	Year	Model
count	4345	4173.000000	4345	4345.000000	4195.000000	4345	4345	4345.000000	4345
unique	7	NaN	6	NaN	NaN	4	2	NaN	312
top	Volkswagen	NaN	sedan	NaN	NaN	Diesel	yes	NaN	E-Class
freq	936	NaN	1649	NaN	NaN	2019	3947	NaN	199
mean	NaN	19418.746935	NaN	161.237284	2.790734	NaN	NaN	2006.550058	NaN
std	NaN	25584.242620	NaN	105.705797	5.066437	NaN	NaN	6.719097	NaN
min	NaN	600.000000	NaN	0.000000	0.600000	NaN	NaN	1969.000000	NaN
25%	NaN	6999.000000	NaN	86.000000	1.800000	NaN	NaN	2003.000000	NaN
50%	NaN	11500.000000	NaN	155.000000	2.200000	NaN	NaN	2008.000000	NaN
75%	NaN	21700.000000	NaN	230.000000	3.000000	NaN	NaN	2012.000000	NaN
max	NaN	300000.000000	NaN	980.000000	99.990000	NaN	NaN	2016.000000	NaN



Dealing with missing values

```
data.isnull().sum()
```

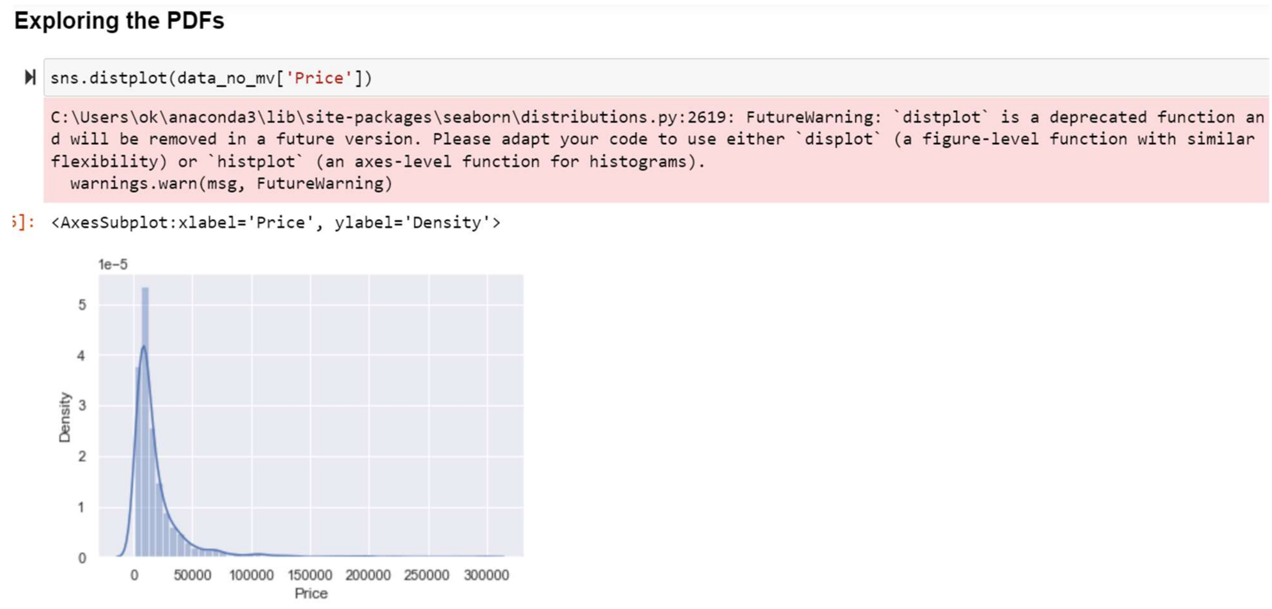
```
3]: Brand             0
    Price           172
    Body              0
    Mileage           0
    EngineV         150
    Engine Type       0
    Registration      0
    Year              0
    Model             0
    dtype: int64
```

```
data_no_mv = data.dropna(axis=0)
```

```
data_no_mv.describe(include='all')
```

5]:

	Brand	Price	Body	Mileage	EngineV	Engine Type	Registration	Year	Model
count	4025	4025.000000	4025	4025.000000	4025.000000	4025	4025	4025.000000	4025
unique	7	NaN	6	NaN	NaN	4	2	NaN	306
top	Volkswagen	NaN	sedan	NaN	NaN	Diesel	yes	NaN	E-Class
freq	880	NaN	1534	NaN	NaN	1861	3654	NaN	188

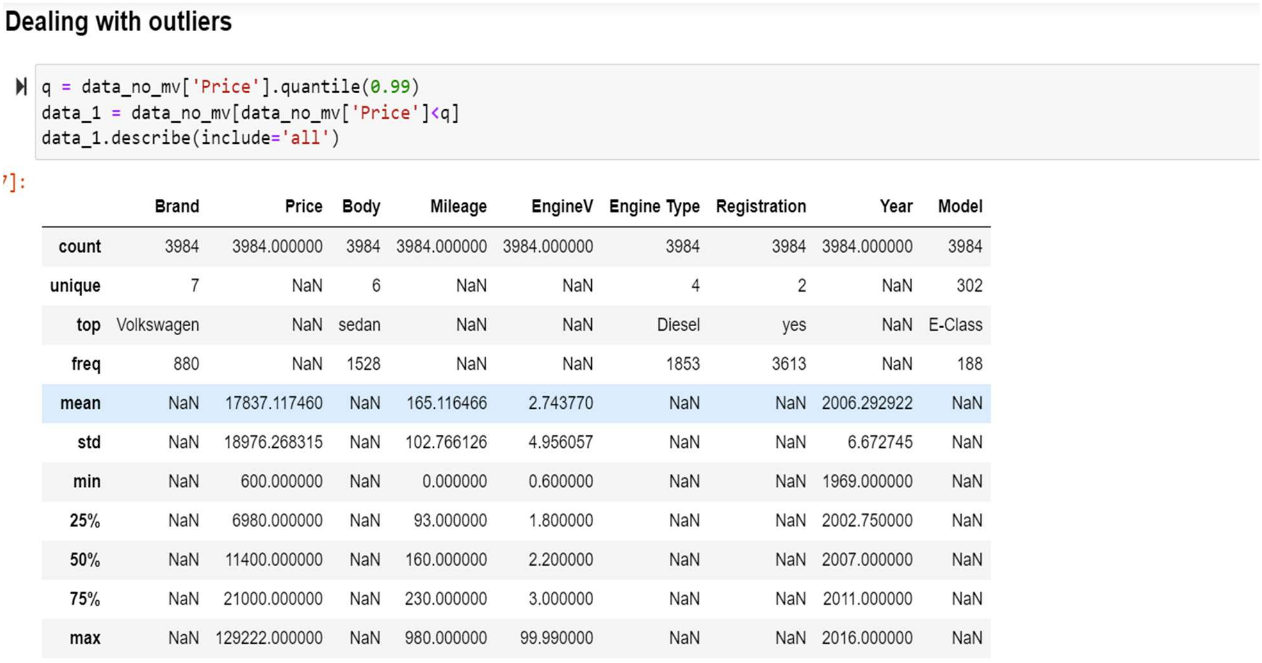


Exploring the PDFs

```
sns.distplot(data_no_mv['Price'])
```

```
C:\Users\ok\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function an
d will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar
flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

```
5]: <AxesSubplot:xlabel='Price', ylabel='Density'>
```

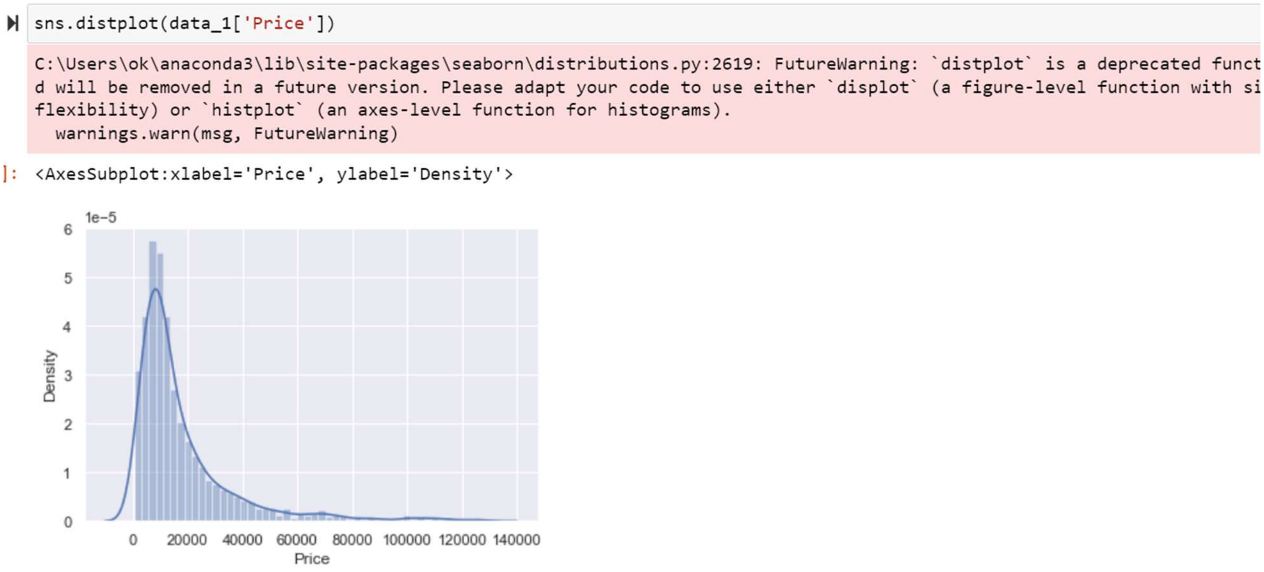


Dealing with outliers

```
q = data_no_mv['Price'].quantile(0.99)
data_1 = data_no_mv[data_no_mv['Price']<q]
data_1.describe(include='all')
```

7]:

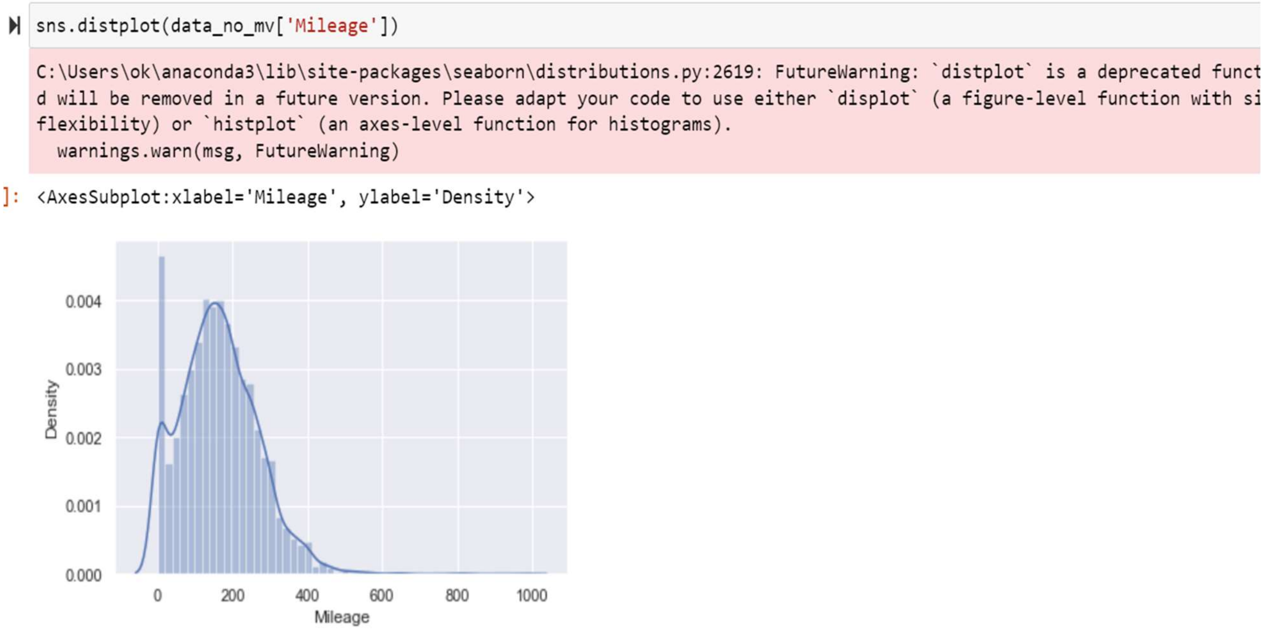
	Brand	Price	Body	Mileage	EngineV	Engine Type	Registration	Year	Model
count	3984	3984.000000	3984	3984.000000	3984.000000	3984	3984	3984.000000	3984
unique	7	NaN	6	NaN	NaN	4	2	NaN	302
top	Volkswagen	NaN	sedan	NaN	NaN	Diesel	yes	NaN	E-Class
freq	880	NaN	1528	NaN	NaN	1853	3613	NaN	188
mean	NaN	17837.117460	NaN	165.116466	2.743770	NaN	NaN	2006.292922	NaN
std	NaN	18976.268315	NaN	102.766126	4.956057	NaN	NaN	6.672745	NaN
min	NaN	600.000000	NaN	0.000000	0.600000	NaN	NaN	1969.000000	NaN
25%	NaN	6980.000000	NaN	93.000000	1.800000	NaN	NaN	2002.750000	NaN
50%	NaN	11400.000000	NaN	160.000000	2.200000	NaN	NaN	2007.000000	NaN
75%	NaN	21000.000000	NaN	230.000000	3.000000	NaN	NaN	2011.000000	NaN
max	NaN	129222.000000	NaN	980.000000	99.990000	NaN	NaN	2016.000000	NaN



```
sns.distplot(data_1['Price'])
```

```
C:\Users\ok\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated funct
d will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with si
flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

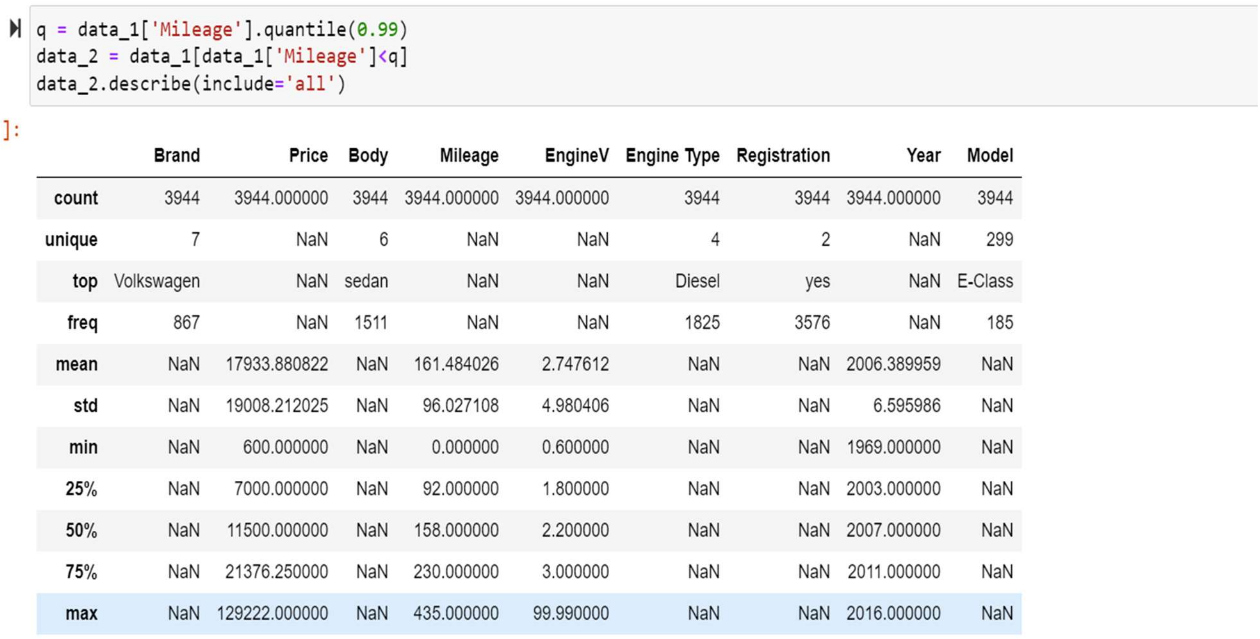
```
]: <AxesSubplot:xlabel='Price', ylabel='Density'>
```



```
sns.distplot(data_no_mv['Mileage'])
```

```
C:\Users\ok\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated funct
d will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with si
flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

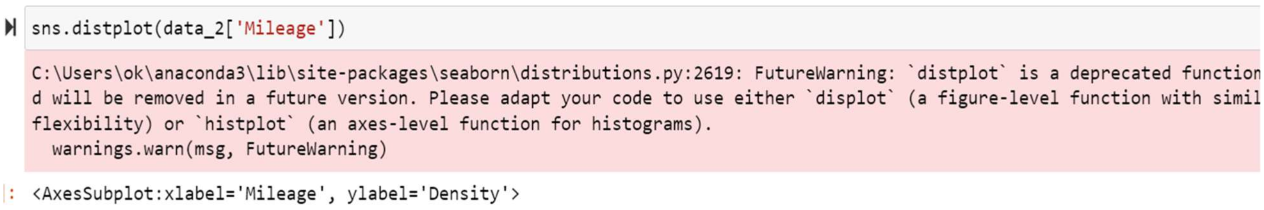
```
]: <AxesSubplot:xlabel='Mileage', ylabel='Density'>
```



```
q = data_1['Mileage'].quantile(0.99)
data_2 = data_1[data_1['Mileage']<q]
data_2.describe(include='all')
```

]:

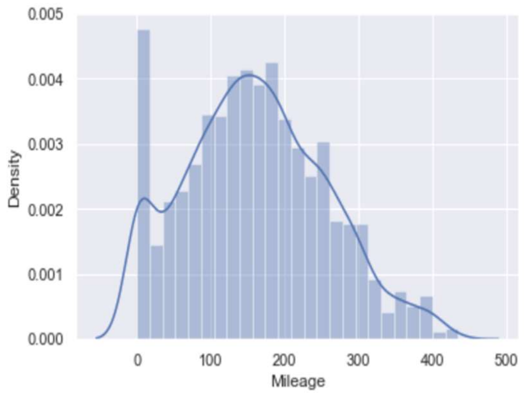
	Brand	Price	Body	Mileage	EngineV	Engine Type	Registration	Year	Model
count	3944	3944.000000	3944	3944.000000	3944.000000	3944	3944	3944.000000	3944
unique	7	NaN	6	NaN	NaN	4	2	NaN	299
top	Volkswagen	NaN	sedan	NaN	NaN	Diesel	yes	NaN	E-Class
freq	867	NaN	1511	NaN	NaN	1825	3576	NaN	185
mean	NaN	17933.880822	NaN	161.484026	2.747612	NaN	NaN	2006.389959	NaN
std	NaN	19008.212025	NaN	96.027108	4.980406	NaN	NaN	6.595986	NaN
min	NaN	600.000000	NaN	0.000000	0.600000	NaN	NaN	1969.000000	NaN
25%	NaN	7000.000000	NaN	92.000000	1.800000	NaN	NaN	2003.000000	NaN
50%	NaN	11500.000000	NaN	158.000000	2.200000	NaN	NaN	2007.000000	NaN
75%	NaN	21376.250000	NaN	230.000000	3.000000	NaN	NaN	2011.000000	NaN
max	NaN	129222.000000	NaN	435.000000	99.990000	NaN	NaN	2016.000000	NaN

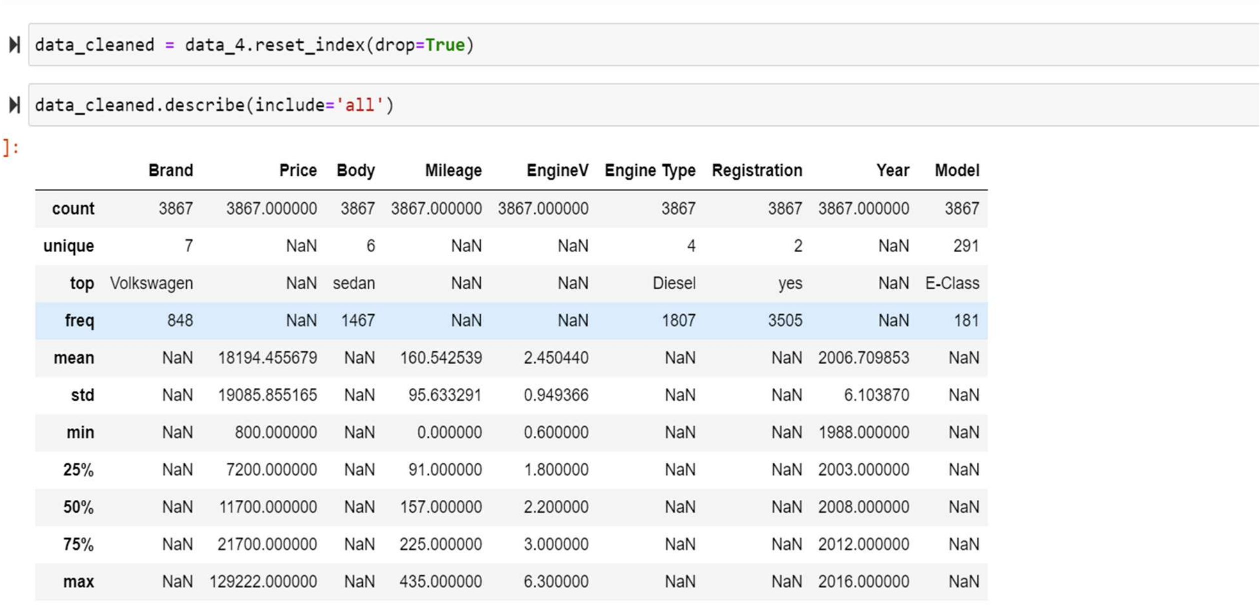


```
sns.distplot(data_2['Mileage'])
```

```
C:\Users\ok\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function
d will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with simil
flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

```
: <AxesSubplot:xlabel='Mileage', ylabel='Density'>
```



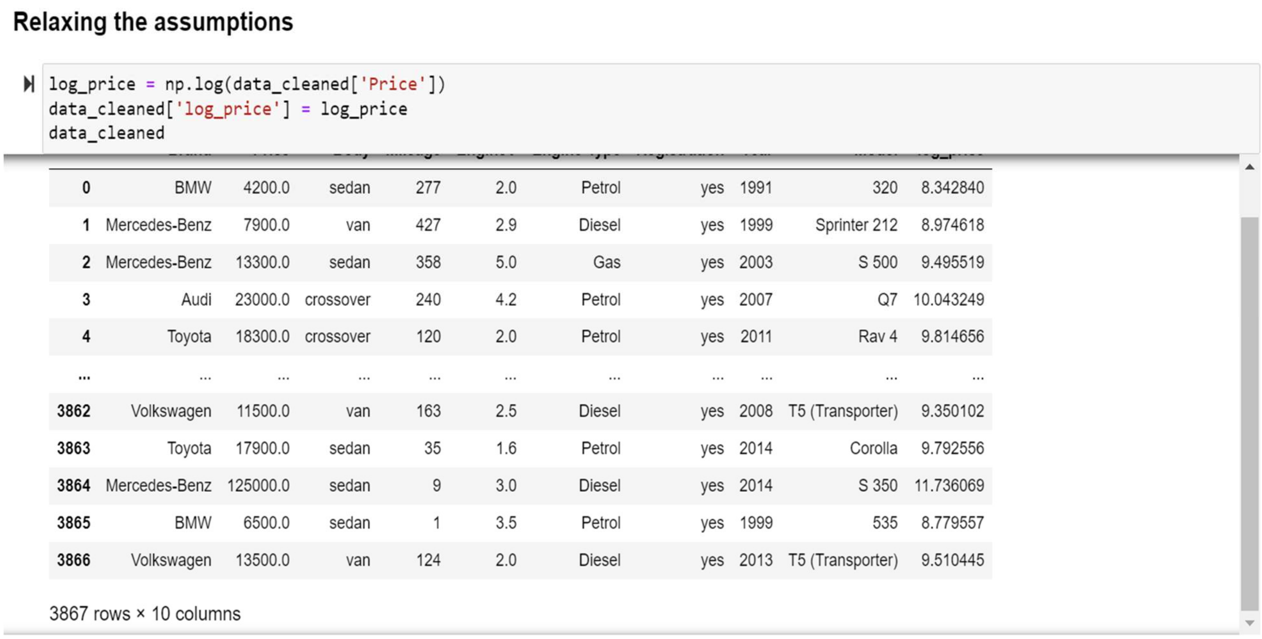


```
data_cleaned = data_4.reset_index(drop=True)
```

```
data_cleaned.describe(include='all')
```

]:

	Brand	Price	Body	Mileage	EngineV	Engine Type	Registration	Year	Model
count	3867	3867.000000	3867	3867.000000	3867.000000	3867	3867	3867.000000	3867
unique	7	NaN	6	NaN	NaN	4	2	NaN	291
top	Volkswagen	NaN	sedan	NaN	NaN	Diesel	yes	NaN	E-Class
freq	848	NaN	1467	NaN	NaN	1807	3505	NaN	181
mean	NaN	18194.455679	NaN	160.542539	2.450440	NaN	NaN	2006.709853	NaN
std	NaN	19085.855165	NaN	95.633291	0.949366	NaN	NaN	6.103870	NaN
min	NaN	800.000000	NaN	0.000000	0.600000	NaN	NaN	1988.000000	NaN
25%	NaN	7200.000000	NaN	91.000000	1.800000	NaN	NaN	2003.000000	NaN
50%	NaN	11700.000000	NaN	157.000000	2.200000	NaN	NaN	2008.000000	NaN
75%	NaN	21700.000000	NaN	225.000000	3.000000	NaN	NaN	2012.000000	NaN
max	NaN	129222.000000	NaN	435.000000	6.300000	NaN	NaN	2016.000000	NaN

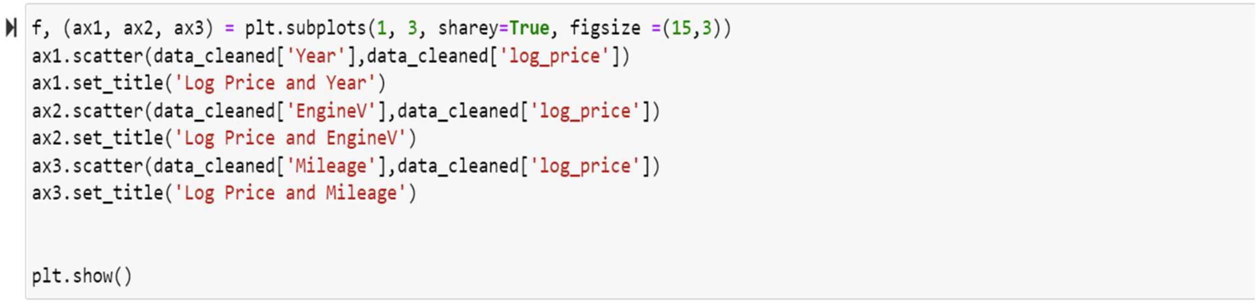


Relaxing the assumptions

```
log_price = np.log(data_cleaned['Price'])
data_cleaned['log_price'] = log_price
data_cleaned
```

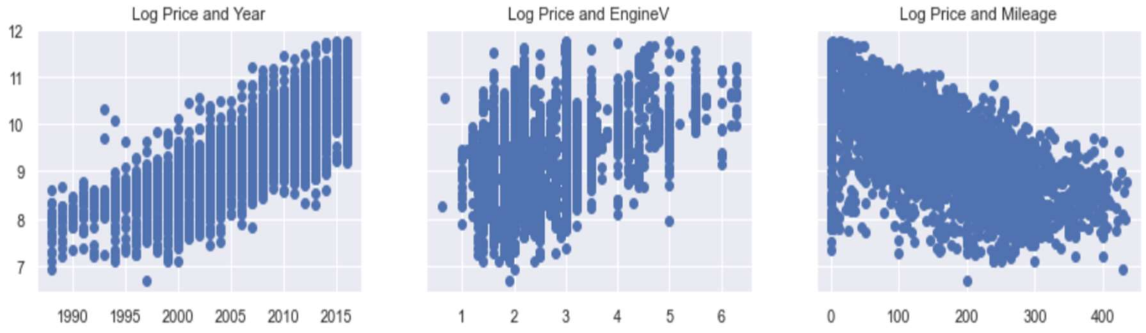
	Brand	Price	Body	Mileage	EngineV	Engine Type	Registration	Year	Model	log_price
0	BMW	4200.0	sedan	277	2.0	Petrol	yes	1991	320	8.342840
1	Mercedes-Benz	7900.0	van	427	2.9	Diesel	yes	1999	Sprinter 212	8.974618
2	Mercedes-Benz	13300.0	sedan	358	5.0	Gas	yes	2003	S 500	9.495519
3	Audi	23000.0	crossover	240	4.2	Petrol	yes	2007	Q7	10.043249
4	Toyota	18300.0	crossover	120	2.0	Petrol	yes	2011	Rav 4	9.814656
...	...	...	...	...	...	...	...	...	...	...
3862	Volkswagen	11500.0	van	163	2.5	Diesel	yes	2008	T5 (Transporter)	9.350102
3863	Toyota	17900.0	sedan	35	1.6	Petrol	yes	2014	Corolla	9.792556
3864	Mercedes-Benz	125000.0	sedan	9	3.0	Diesel	yes	2014	S 350	11.736069
3865	BMW	6500.0	sedan	1	3.5	Petrol	yes	1999	535	8.779557
3866	Volkswagen	13500.0	van	124	2.0	Diesel	yes	2013	T5 (Transporter)	9.510445

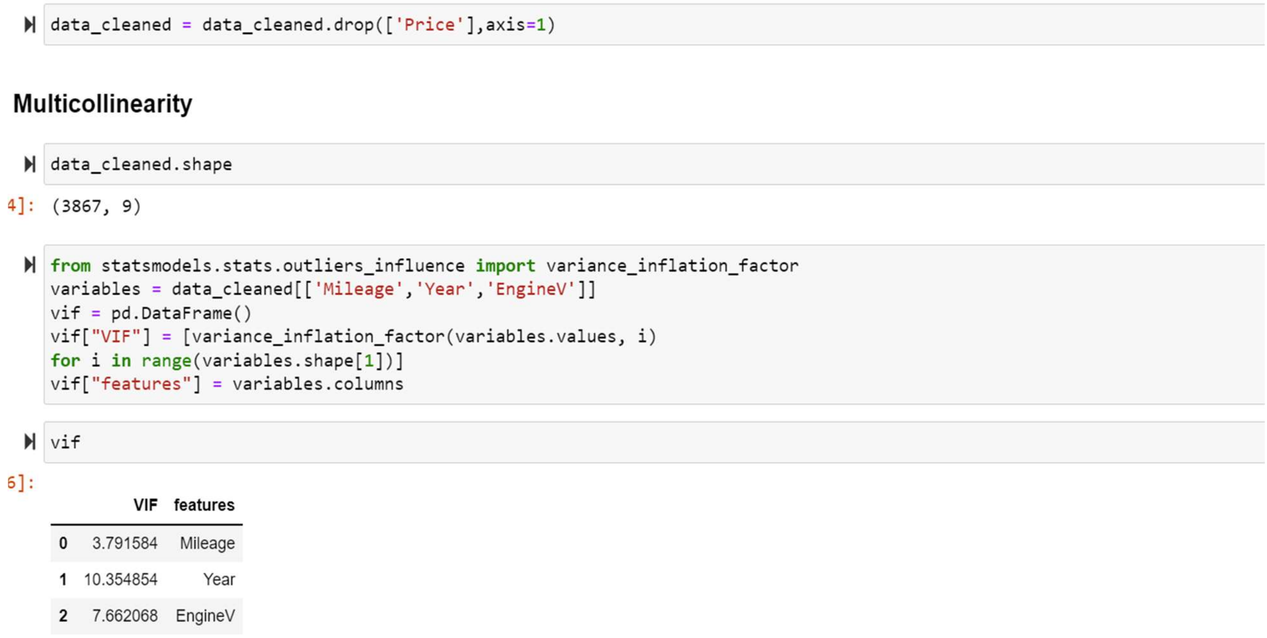
3867 rows × 10 columns



```
f, (ax1, ax2, ax3) = plt.subplots(1, 3, sharey=True, figsize =(15,3))
ax1.scatter(data_cleaned['Year'],data_cleaned['log_price'])
ax1.set_title('Log Price and Year')
ax2.scatter(data_cleaned['EngineV'],data_cleaned['log_price'])
ax2.set_title('Log Price and EngineV')
ax3.scatter(data_cleaned['Mileage'],data_cleaned['log_price'])
ax3.set_title('Log Price and Mileage')


plt.show()
```





```
data_cleaned = data_cleaned.drop(['Price'],axis=1)
```

Multicollinearity

```
data_cleaned.shape
```

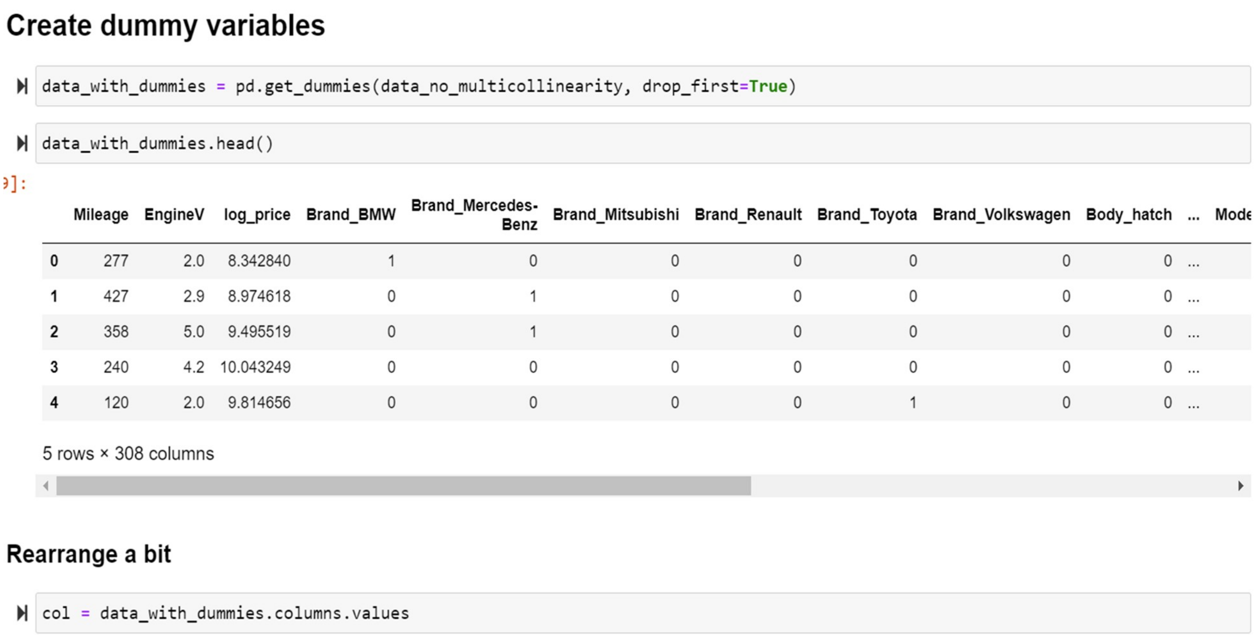
```
4]: (3867, 9)
```

```
from statsmodels.stats.outliers_influence import variance_inflation_factor
variables = data_cleaned[['Mileage','Year','EngineV']]
vif = pd.DataFrame()
vif["VIF"] = [variance_inflation_factor(variables.values, i)
for i in range(variables.shape[1])]
vif["features"] = variables.columns
```

```
vif
```

6]:

	VIF	features
0	3.791584	Mileage
1	10.354854	Year
2	7.662068	EngineV



Create dummy variables

```
data_with_dummies = pd.get_dummies(data_no_multicollinearity, drop_first=True)
```

```
data_with_dummies.head()
```

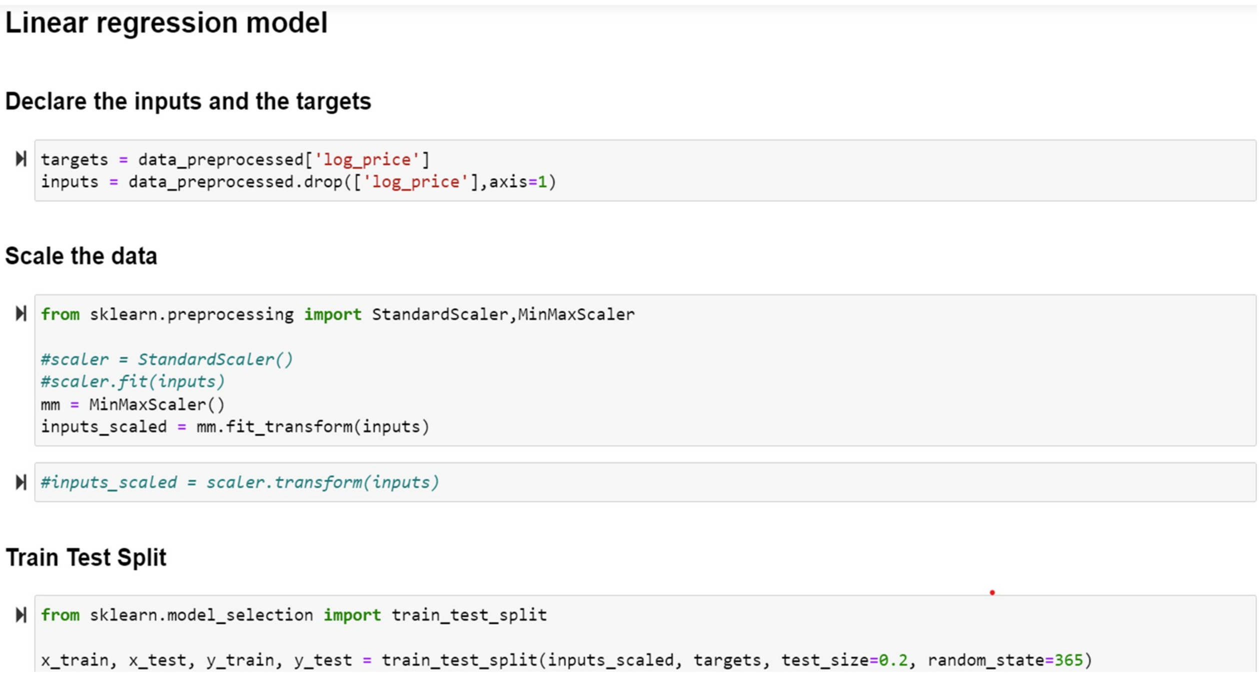
]:

	Mileage	EngineV	log_price	Brand_BMW	Brand_Mercedes-Benz	Brand_Mitsubishi	Brand_Renault	Brand_Toyota	Brand_Volkswagen	Body_hatch	...	Mode
0	277	2.0	8.342840	1	0	0	0	0	0	0	...	
1	427	2.9	8.974618	0	1	0	0	0	0	0	...	
2	358	5.0	9.495519	0	1	0	0	0	0	0	...	
3	240	4.2	10.043249	0	0	0	0	0	0	0	...	
4	120	2.0	9.814656	0	0	0	0	1	0	0	...	

5 rows × 308 columns

Rearrange a bit

```
col = data_with_dummies.columns.values
```



Linear regression model

Declare the inputs and the targets

```
targets = data_preprocessed['log_price']
inputs = data_preprocessed.drop(['log_price'],axis=1)
```

Scale the data

```
from sklearn.preprocessing import StandardScaler,MinMaxScaler

#scaler = StandardScaler()
#scaler.fit(inputs)
mm = MinMaxScaler()
inputs_scaled = mm.fit_transform(inputs)
```

```
#inputs_scaled = scaler.transform(inputs)
```

Train Test Split

```
from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(inputs_scaled, targets, test_size=0.2, random_state=365)
```