import libraries

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.tree import DecisionTreeRegressor
```

In [2]:

```
# data loading

dataset = pd.read_csv('petrol_consumption.csv')
dataset.head()
```

Out[2]:

	Petrol_tax	Average_income	Paved_Highways	Population_Driver_licence(%)	Petrol_Consu
0	9.0	3571	1976	0.525	
1	9.0	4092	1250	0.572	
2	9.0	3865	1586	0.580	
3	7.5	4870	2351	0.529	
4	8.0	4399	431	0.544	

In [3]:

```
dataset.shape
```

Out[3]:

```
(48, 5)
```

In [4]:

```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 48 entries, 0 to 47
Data columns (total 5 columns):
 #   Column                         Non-Null Count  Dtype  
---  ------                         --------------  -----  
 0   Petrol_tax                     48 non-null     float64
 1   Average_income                 48 non-null     int64  
 2   Paved_Highways                 48 non-null     int64  
 3   Population_Driver_licence(%)  48 non-null     float64
 4   Petrol_Consumption             48 non-null     int64  
dtypes: float64(2), int64(3)
memory usage: 2.0 KB
```

In [5]:
```
dataset.describe()
```

Out[5]:

	Petrol_tax	Average_income	Paved_Highways	Population_Driver_licence(%)	Petrol_C
count	48.000000	48.000000	48.000000	48.000000	
mean	7.668333	4241.833333	5565.416667	0.570333	
std	0.950770	573.623768	3491.507166	0.055470	
min	5.000000	3063.000000	431.000000	0.451000	
25%	7.000000	3739.000000	3110.250000	0.529750	
50%	7.500000	4298.000000	4735.500000	0.564500	
75%	8.125000	4578.750000	7156.000000	0.595250	
max	10.000000	5342.000000	17782.000000	0.724000	

In [8]:
```
# seperate out features and target value from dataset

X = dataset.drop(['Petrol_Consumption'],axis = 1).values
y = dataset['Petrol_Consumption'].values
```

In [9]:
```
X.shape
```

Out[9]:
```
(48, 4)
```

In [10]:
```
y.shape
```

Out[10]:
```
(48,)
```

In [35]:
```
# split the data in training and testing set

X_train, X_test, y_train,y_test = train_test_split(X,y, test_size =
0.1, random_state = 42)
```

In [36]:
```
# Model

reg = DecisionTreeRegressor()

# fitting
reg.fit(X_train,y_train)
```

Out[36]:
```
DecisionTreeRegressor()
```

In [37]:
```
# predicting
y_pred = reg.predict(X_test)
y_pred
```

Out[37]:
```
array([603., 525., 574., 699., 498.])
```

In [40]:

```
# calculate RMSE

rmse = np.sqrt(metrics.mean_squared_error(y_test,y_pred))
print(rmse)
```

```
59.57348403442592
```

In [30]:

```
y_pred_df = pd.DataFrame(y_pred)
y_pred_df
```

Out[30]:

	0
0	644.0
1	525.0
2	574.0
3	632.0
4	510.0

In [31]:

```
y_pred_df['actual'] = y_test
```

In [32]:

```
y_pred_df
```

Out[32]:

	0	actual
0	644.0	631
1	525.0	587
2	574.0	577
3	632.0	591
4	510.0	460

In [33]:

```
y_pred_df.columns = ['Predicted', 'actual']
```

In [34]:

```
y_pred_df
```

Out[34]:

	Predicted	actual
0	644.0	631
1	525.0	587
2	574.0	577
3	632.0	591
4	510.0	460

In [ ]:

```
```

```
In [ ]:
```