GAYATRI VIDYA PARISHAD COLLEGE OF ENGINEERING (Autonomous)
Approved by AICTE & Affiliated to Andhra University, Visakhapatnam from 2022-23
(Affiliated to JNTUK, Kakinada upto 2021-22)
Re-accredited by NAAC twice with 'A' Grade with a CGPA of 3.47/4.00
Madhurawada, Visakhapatnam - 530048

20CS1109 - Machine Learning Applications Lab

Lab Record :WEEK-13

Name: ESWAR

Roll No. : 20131A05Q9

Department: Computer Science and Engineering

Section: 4

AIM:

Write a program to construct a Regression tree for cost estimation by assuming any numerical dataset.

Code:

[]: from google.colab import drive

drive.mount('/content/drive')

O/P: Mounted at /content/drive

[]: import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

df = pd.read_csv('cars.csv')

df

O/P:

	selling_price	year	km_driven	mileage	engine	max_power	age	Diesel	Electric	LPG	Petrol	Manual	5seats	>5seats
0	1.20	2012.0	120000	19.70	796.0	46.30	10.0	0	0	0	1	1	1	0
1	5.50	2016.0	20000	18.90	1197.0	82.00	6.0	0	0	0	1	1	1	0
2	2.15	2010.0	60000	17.00	1197.0	80.00	12.0	0	0	0	1	1	1	0
3	2.26	2012.0	37000	20.92	998.0	67.10	10.0	0	0	0	1	1	1	0
4	5.70	2015.0	30000	22.77	1498.0	98.59	7.0	1	0	0	0	1	1	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
19815	6.50	2017.0	69480	23.59	1364.0	67.05	5.0	1	0	0	0	1	1	0
19816	9.25	2019.0	18000	17.50	1373.0	91.10	3.0	0	0	0	1	1	0	1
19817	4.25	2015.0	67000	21.14	1498.0	103.52	7.0	1	0	0	0	1	1	0
19818	12.25	2016.0	3800000	16.00	2179.0	140.00	6.0	1	0	0	0	1	0	1
19819	12.00	2019.0	13000	18.00	1497.0	117.60	3.0	0	0	0	1	0	1	0

19820 rows × 14 columns

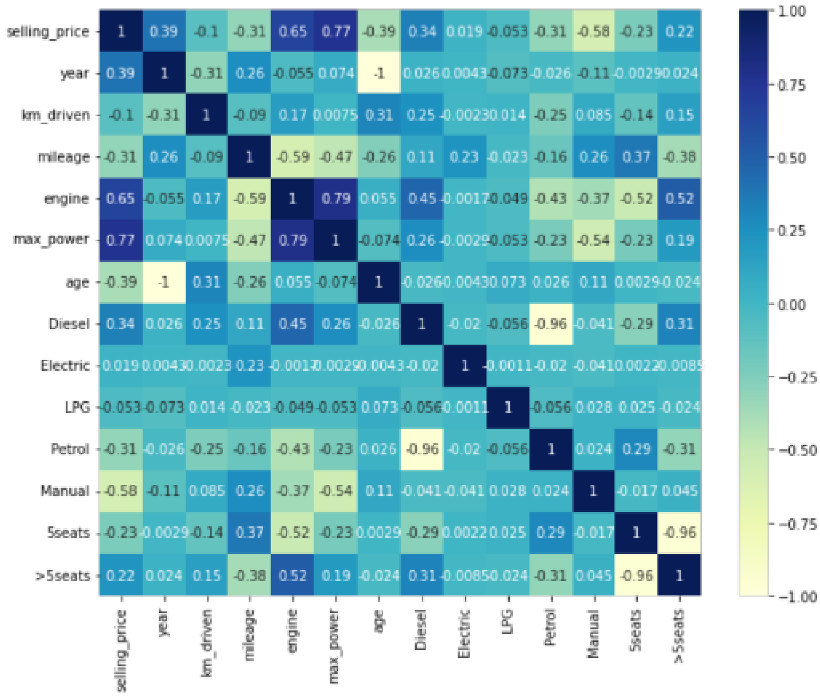
[]: df.describe()

	selling_price	year	km_driven	mileage	engine	max_power	age	Diesel	Electric	LPG	Petrol	Manual	5seats	>5seats
count	19820.000000	19820.000000	1.982000e+04	19820.000000	19820.000000	19820.000000	19820.000000	19820.000000	19820.000000	19820.000000	19820.000000	19820.000000	19820.000000	19820.000000
mean	6.585509	2014.561453	5.815856e+04	19.503402	1475.702381	98.122907	7.438547	0.492583	0.000404	0.003229	0.487841	0.802674	0.835015	0.152825
std	4.847364	3.196636	5.171563e+04	4.297784	518.571223	44.761727	3.196636	0.499958	0.020087	0.056734	0.499865	0.397990	0.371176	0.359828
min	0.300000	1992.000000	1.000000e+02	4.000000	0.000000	5.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	3.410000	2013.000000	3.100000e+04	16.950000	1197.000000	73.900000	5.000000	0.000000	0.000000	0.000000	0.000000	1.000000	1.000000	0.000000
50%	5.200000	2015.000000	5.200000e+04	19.300000	1248.000000	86.800000	7.000000	0.000000	0.000000	0.000000	0.000000	1.000000	1.000000	0.000000
75%	7.850000	2017.000000	7.400000e+04	22.320000	1582.000000	112.000000	9.000000	1.000000	0.000000	0.000000	1.000000	1.000000	1.000000	0.000000
max	20.902500	2021.000000	3.800000e+06	120.000000	6752.000000	626.000000	30.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

[]: plt.figure(figsize=(10,8))

sns.heatmap(df.corr(),cmap = "YlGnBu",annot=True)

O/P:



[]: from sklearn.model_selection import train_test_split

X = df.drop('selling_price', axis=1)

Y = df['selling_price']

X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.3, random_state=1)

[]: from sklearn.linear_model import DecisionTreeRegressor

created the model

dtr = DecisionTreeRegressor(max_depth=5)

training of the model

dtr.fit(X_train, y_train)

O/P:

```
▾ DecisionTreeRegressor
DecisionTreeRegressor(max_depth=5)
```

[]: # predicting on the test data.

y_pred = model.predict(X_test)

y_pred[:15].round(decimals=2)

O/P: array([-0.99, 5.05, 6.12, 6.66, 4.8 , 4.9 , 3.76, 5.62, 6.72, 16.95, 10.1 , 5.78,
0.91, 6.19, 4.64])

```
[]: # actual price of test/new cars.
y_test[:15]
```

O/P:

```
 14690 1.35
134 4.15
4982 4.90
11940 3.31
10861 5.80
2934 3.55
2675 3.50
17621 3.10
14638 6.00
10366 9.50
12507 8.99
6060 6.50
16111 1.90
17240 5.75
7632 2.30
Name: selling_price, dtype: float64
```

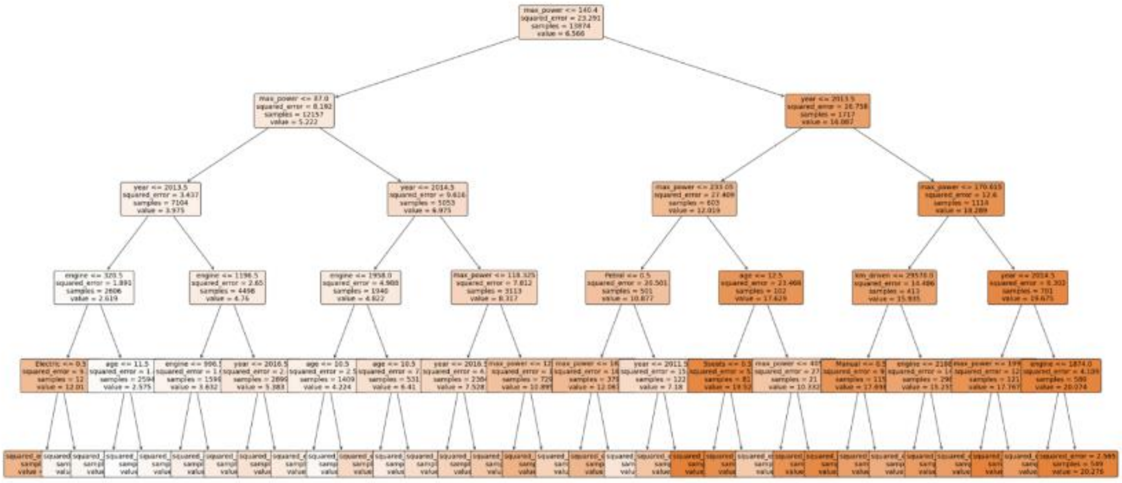
```
[]: print("Model Score on Training Data =",dtr.score(X_train, y_train))
print("Model Score on Test Data =",dtr.score(X_test, y_test))
```

O/P: Model Score on Training Data = 0.9983208637496211

Model Score on Test Data = 0.8974707189363473

```
[]: import matplotlib.pyplot as plt
from sklearn.tree import plot_tree
plt.figure(figsize=(40,20))
a = plot_tree(dtr, feature_names=X.columns, class_names=Y, filled=True,
rounded=True, fontsize=14)
```

O/P:



```
[]: c = df.corr()
f = c[abs(c['selling_price']) > 0.3].index.tolist()
f.remove('selling_price')
df1 = df[f + ['selling_price']]
df1.head()
```

O/p:

	year	mileage	engine	max_power	age	Diesel	Petrol	Manual	selling_price
0	2012.0	19.70	796.0	46.30	10.0	0	1	1	1.20
1	2016.0	18.90	1197.0	82.00	6.0	0	1	1	5.50
2	2010.0	17.00	1197.0	80.00	12.0	0	1	1	2.15
3	2012.0	20.92	998.0	67.10	10.0	0	1	1	2.26
4	2015.0	22.77	1498.0	98.59	7.0	1	0	1	5.70

```
[]: a = df1.drop('selling_price',axis=1)
b = df1['selling_price']
a_train, a_test, b_train, b_test = train_test_split(a, b, test_size=0.3, random_state=1)
r = DecisionTreeRegressor(max_depth = 5)
d = r.fit(a_train,b_train)
```

```
[]: print("Model Score on Training Data =",d.score(a_train,b_train))
print("Model Score on Test Data =",d.score(a_test,b_test))
```

O/P: Model Score on Training Data = 0.8630828722899455

Model Score on Test Data = 0.8577358652461252

[]: b_pred = d.predict(a_test)

plt.scatter(b_test, b_pred)

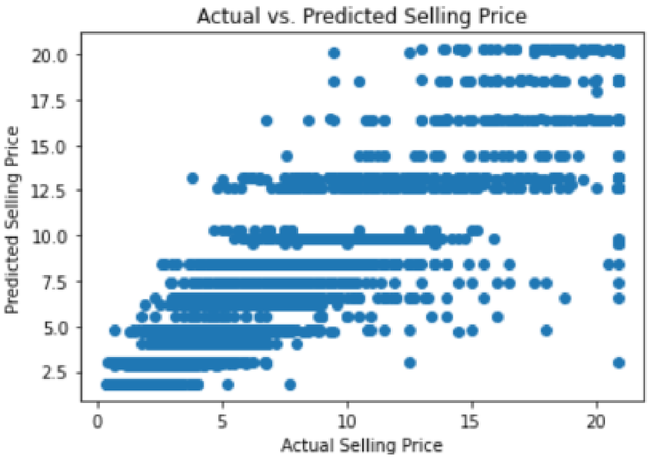
plt.xlabel("Actual Selling Price")

plt.ylabel("Predicted Selling Price")

plt.title("Actual vs. Predicted Selling Price")

plt.show()

O/P:



[]: from sklearn.metrics import mean_squared_error

mse = mean_squared_error(b_test, b_pred)

print("Mean Squared Error:", mse)

O/P: Mean Squared Error: 3.4100997415362766