SUPERVISED ML - REGRESSION MODELS

Problem Statement and

Car price prediction : To predict the selling price of used cars with linear machine learning models such as Multiple linear regression, Decision tree, Random forest, Support vector machine and Polynomial regression.

About the Dataset

To predict the Selling price (Target variable - dependent_variables) of the used cars with features like - Car name, years, kms driven, present price, fuel type, seller type, transmission and owner (Predictors - independent_variables).

```
data = pd.read_csv("cars.csv")
data.head()
```

✓ 0.0s

	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner
0	ritz	2014	3.35	5.59	27000	Petrol	Dealer	Manual	0
1	sx4	2013	4.75	9.54	43000	Diesel	Dealer	Manual	0
2	ciaz	2017	7.25	9.85	6900	Petrol	Dealer	Manual	0
3	wagon r	2011	2.85	4.15	5200	Petrol	Dealer	Manual	0
4	swift	2014	4.60	6.87	42450	Diesel	Dealer	Manual	0

View of the dataset

Steps involved in the prediction process

1) Import the required libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.svm import SVR
from sklearn.preprocessing import PolynomialFeatures, LabelEncoder, StandardScaler
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import mean_squared_error
from scipy import stats
from scipy.stats import norm
from dataprep.eda import create_report

import warnings
warnings.filterwarnings("ignore")
```

✓ 0.0s

2) Exploratory data analysis (EDA)

- Shape of the data

```
data.shape
✓ 0.0s
(301, 9)
```

- Info about the dataset about the columns and its data types

```
data.info()
✓ 0.0s
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 301 entries, 0 to 300
Data columns (total 9 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Car_Name       301 non-null    object
 1   Year           301 non-null    int64
 2   Selling_Price  301 non-null    float64
 3   Present_Price  301 non-null    float64
 4   Kms_Driven     301 non-null    int64
 5   Fuel_Type      301 non-null    object
 6   Seller_Type    301 non-null    object
 7   Transmission   301 non-null    object
 8   Owner          301 non-null    int64
dtypes: float64(2), int64(3), object(4)
memory usage: 21.3+ KB
```

- Description of numerical data like mean, standard deviation, minimum value, maximum value and 25th, 50th and 75th percentile.

```
data.describe()
```

✓ 0.0s

	Year	Selling_Price	Present_Price	Kms_Driven	Owner
count	301.000000	301.000000	301.000000	301.000000	301.000000
mean	2013.627907	4.661296	7.628472	36947.205980	0.043189
std	2.891554	5.082812	8.644115	38886.883882	0.247915
min	2003.000000	0.100000	0.320000	500.000000	0.000000
25%	2012.000000	0.900000	1.200000	15000.000000	0.000000
50%	2014.000000	3.600000	6.400000	32000.000000	0.000000
75%	2016.000000	6.000000	9.900000	48767.000000	0.000000
max	2018.000000	35.000000	92.600000	500000.000000	3.000000

- To check the missing values in the dataset

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

✓ 0.0s

```
Car_Name          0
Year              0
Selling_Price     0
Present_Price     0
Kms_Driven        0
Fuel_Type         0
Seller_Type       0
Transmission      0
Owner             0
dtype: int64
```

- Stats report generated

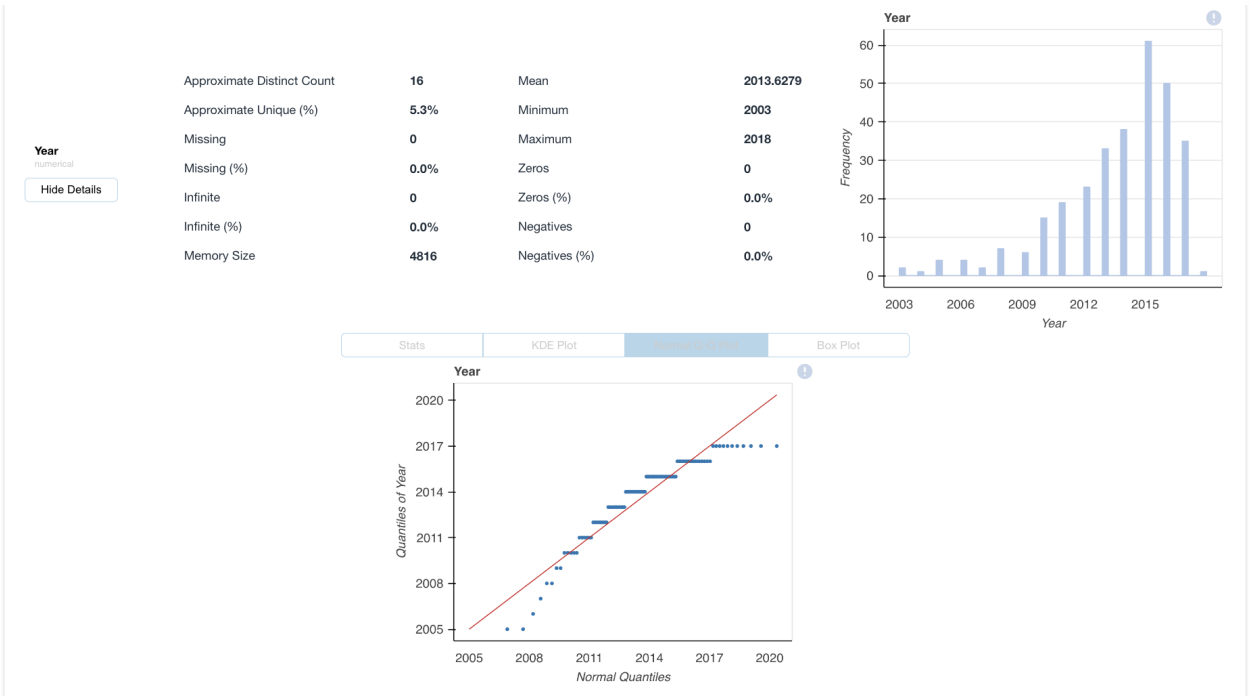
Overview

Dataset Statistics

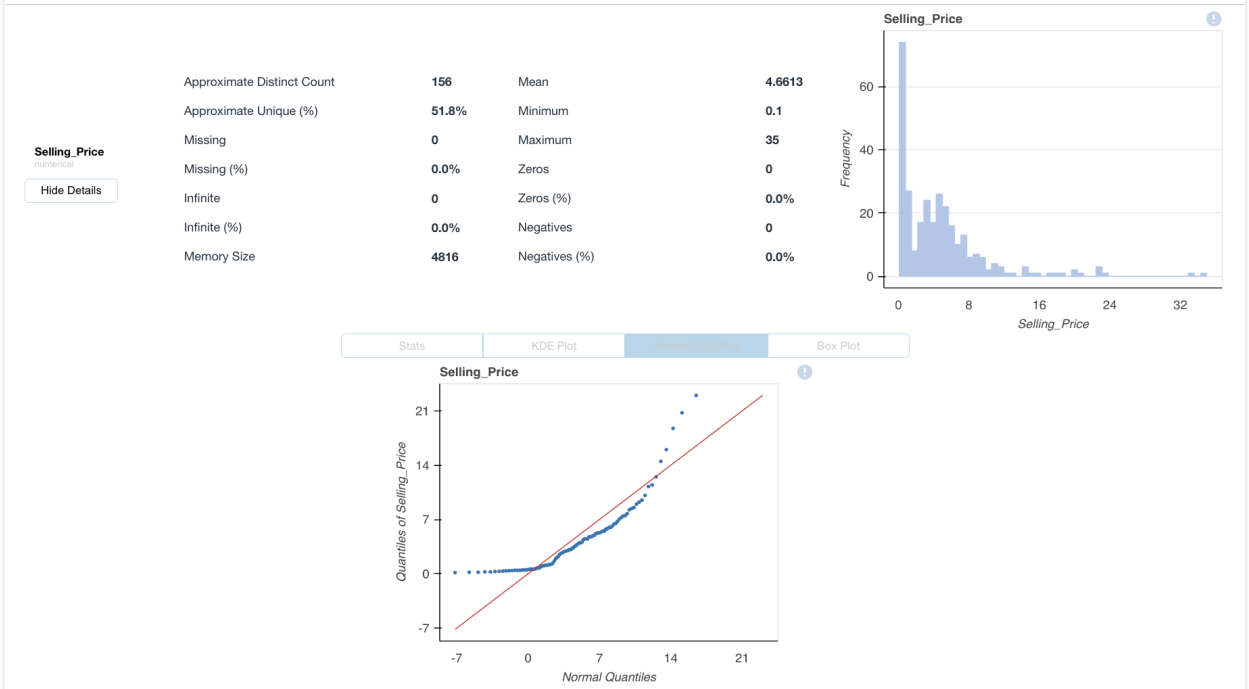
Number of Variables	9
Number of Rows	301
Missing Cells	0
Missing Cells (%)	0.0%
Duplicate Rows	2
Duplicate Rows (%)	0.7%
Total Size in Memory	87.6 KB
Average Row Size in Memory	298.1 B
Variable Types	Categorical: 5 Numerical: 4

Dataset Insights

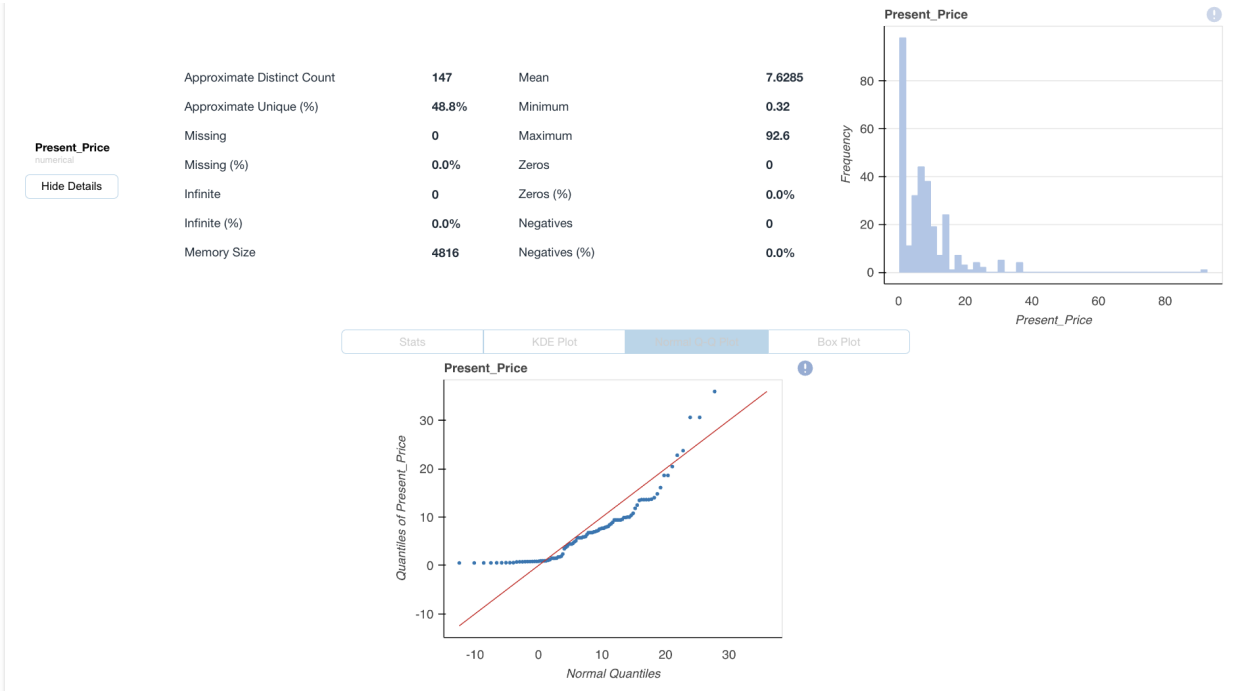
`Year` is skewed	Skewed
`Sellin…` is skewed	Skewed
`Presen…` is skewed	Skewed
`Kms_Dr…` is skewed	Skewed
`Car_Na…` has a high cardinality: 98 distinct values	High Cardinality
`Owner` has constant length 1	Constant Length



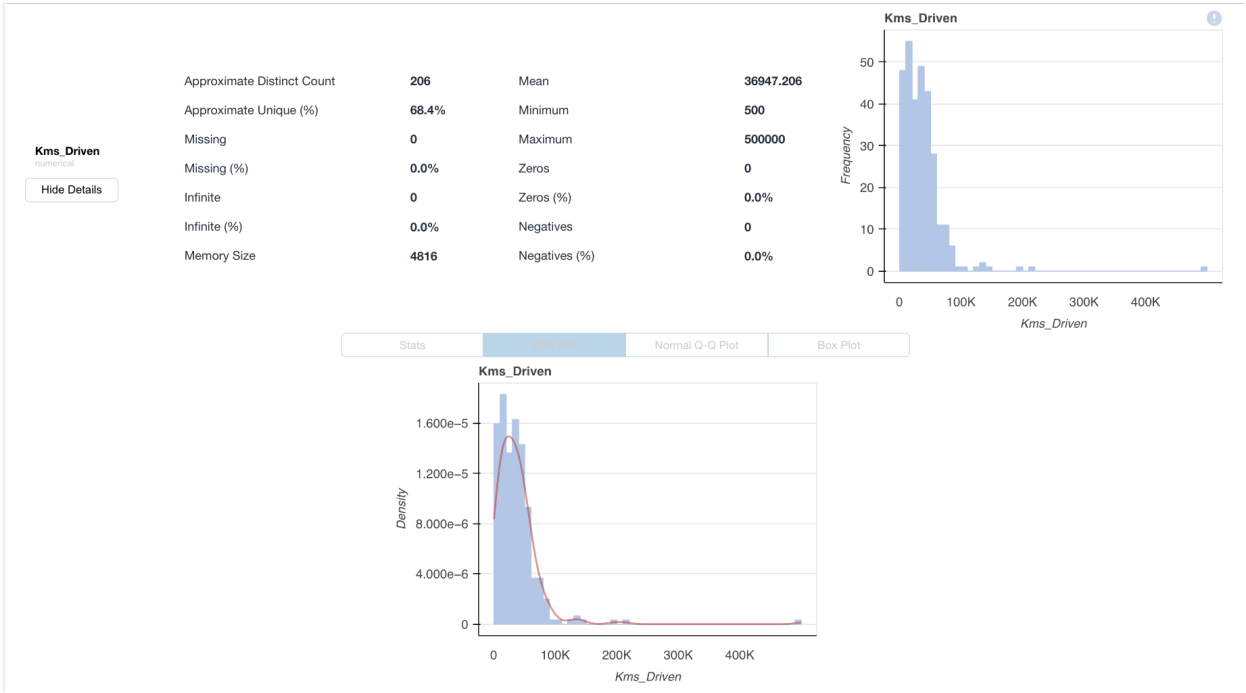
*Variable - **Year** almost in normal distribution and left skewed*



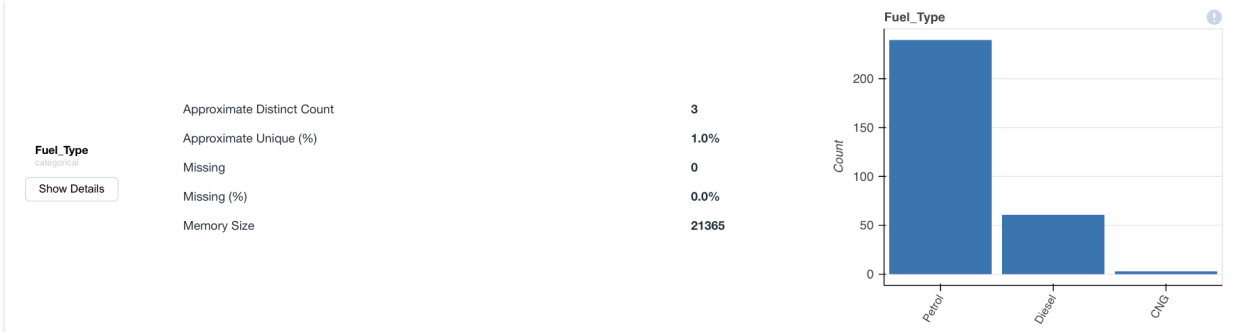
*Variable - **Selling_price***



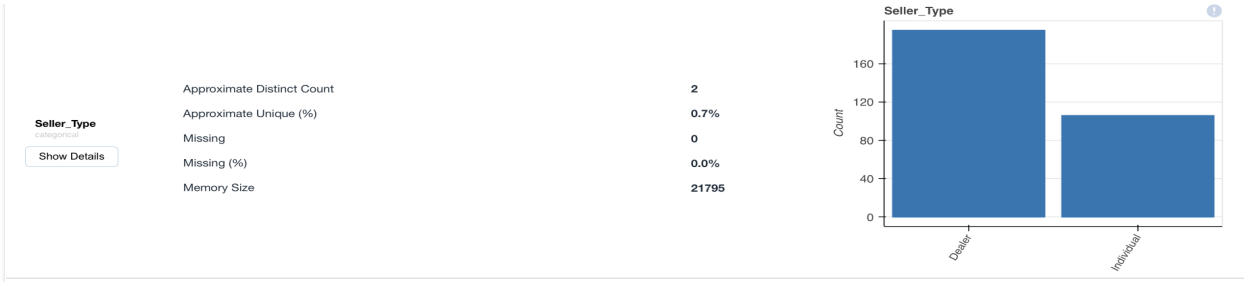
*Variable - **Present_price** right skewed (The price at which the cars are selled lies from 0 to 20 lakhs)*



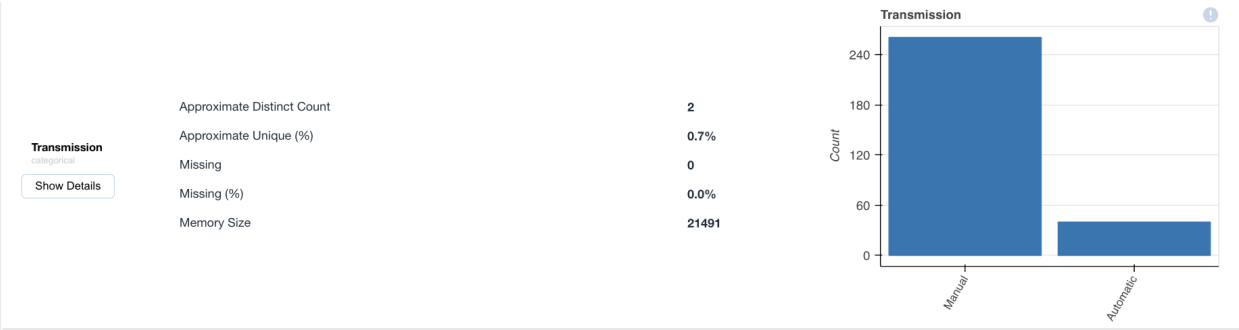
*Variable - **Kms_Driven** This is also right skewed the majority of data lies from 0 to 100k kms.*



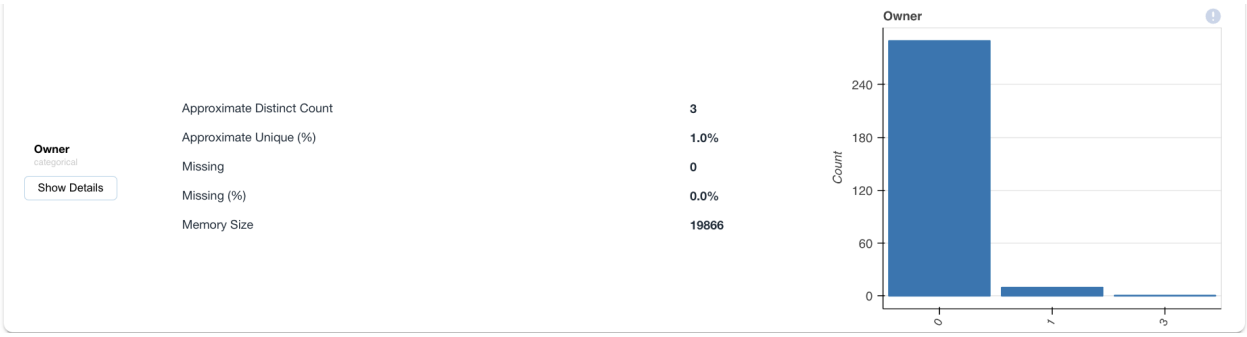
*Variable **Fuel_type** consists of three types petrol, Diesel and CNG with most petrol cars used.*



*Variable **Seller_type** consists of types Dealer and Individual.*

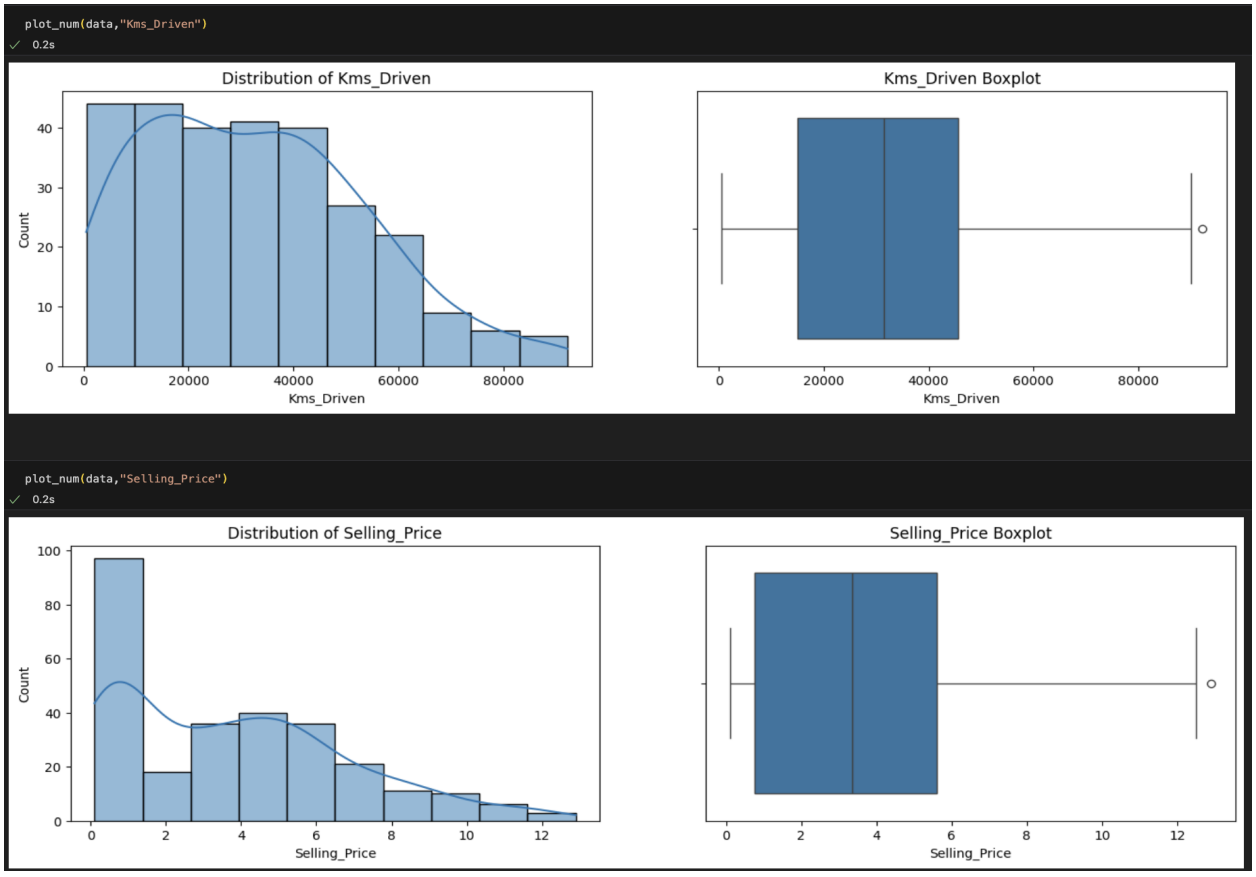


*Variable **Transmission** consists of Manual and Automatic.*

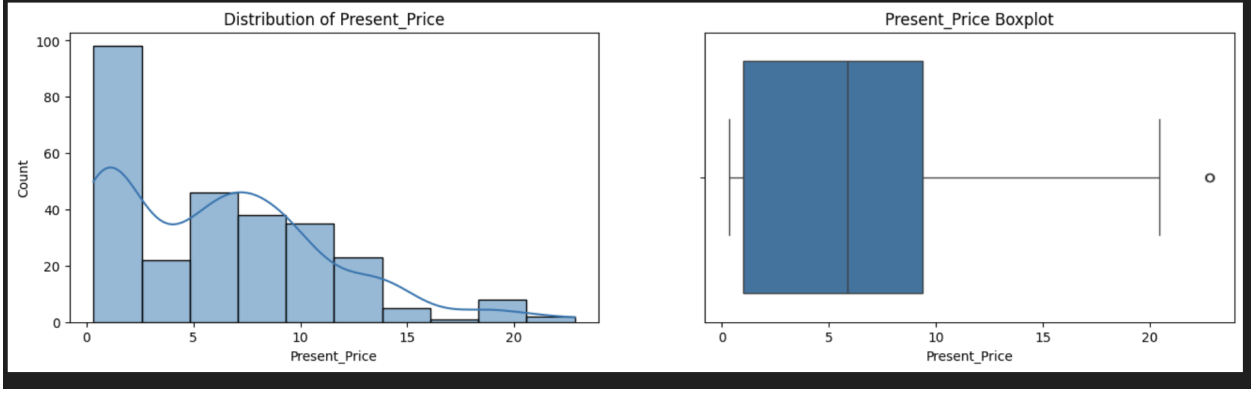


*Variable **Owner** consists of 0, 1 and 2. Where cars with 0 owners are high.*

- **Outliers Removal** - the outliers of numerical variables are almost removed with the help of IQR range.



```
plot_num(data,"Present_Price")
✓ 0.2s
```



3) Data Preprocessing

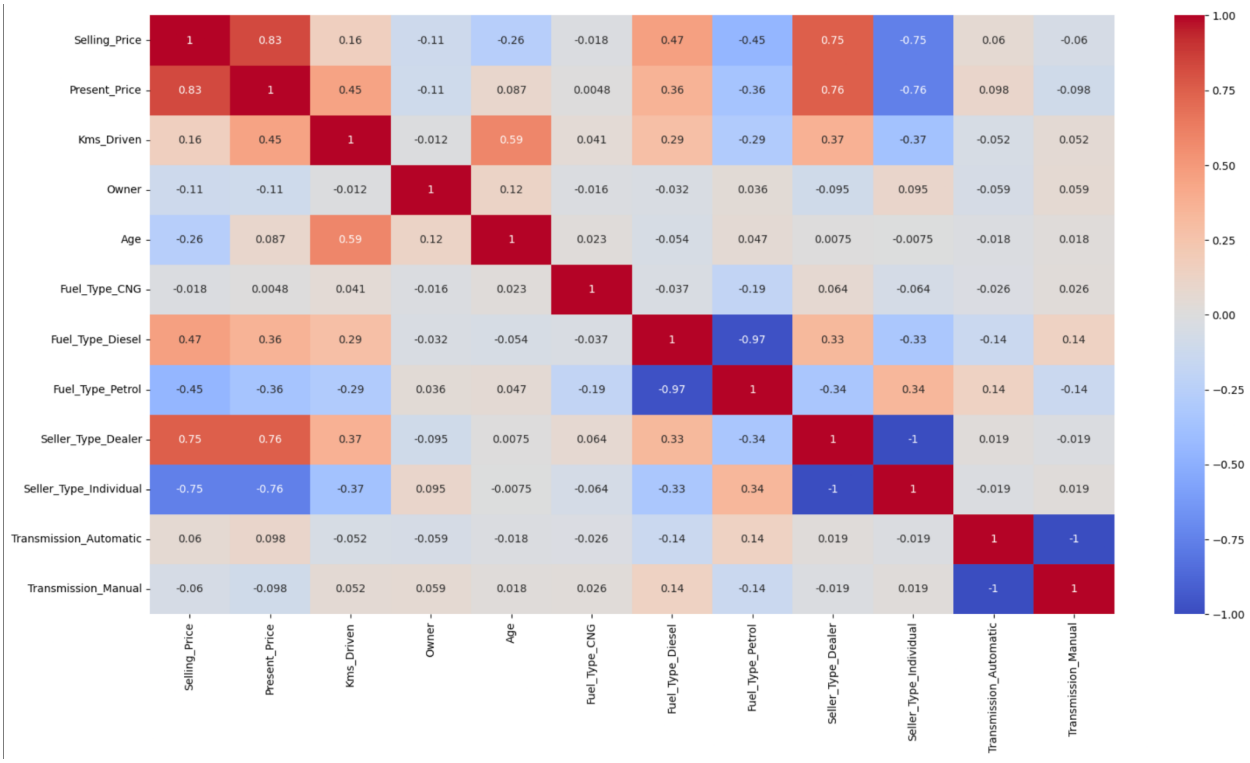
- **Feature Engineering** - The age of the car is calculated with the use of column year subtracted with 2024.
- The features **car_name** and **year** are dropped from the dataset.
- The categorical features are converted into numerical with the help of one_hot_encoding.

```
data = pd.get_dummies(data, columns=categorical_cols)
data
✓ 0.0s                                                                    Python
```

	Selling_Price	Present_Price	Kms_Driven	Owner	Age	Fuel_Type_CNG	Fuel_Type_Diesel	Fuel_Type_Petrol	Seller_Type_Dealer	Seller_Type_Individual	Transmissio
0	3.35	5.59	27000	0	10	0	0	1	1	0	
1	4.75	9.54	43000	0	11	0	1	0	1	0	
2	7.25	9.85	6900	0	7	0	0	1	1	0	
3	2.85	4.15	5200	0	13	0	0	1	1	0	
4	4.60	6.87	42450	0	10	0	1	0	1	0	
...	...	...	...	...	...	...	...	...	...	...	
296	9.50	11.60	33988	0	8	0	1	0	1	0	
297	4.00	5.90	60000	0	9	0	0	1	1	0	
298	3.35	11.00	87934	0	15	0	0	1	1	0	
299	11.50	12.50	9000	0	7	0	1	0	1	0	
300	5.30	5.90	5464	0	8	0	0	1	1	0	

278 rows x 12 columns

- Heatmap - Correlation between every variable is shown.



- **Scaling** - The variables are scaled with a standard scaler which uses z-score to scale the values to a specific -ve and +ve range.
- The data is separated into X and Y data and splitted into training and testing sets.

4) Model training

- The models are trained and the metrics like Root mean squared error, accuracy of testing and training sets are calculated.

```
models
✓ 0.0s
```

	model	RMSE	ATrS	ATeS
0	MultipleLinearRegressor	0.500838	0.877828	0.792114
1	DecisionTreeRegressor	0.454782	0.887030	0.828589
2	RandomForestRegressor	0.297807	0.993653	0.926498
3	SupportVectorRegressor	0.310413	0.954128	0.920143
4	PolynomialRegressor	0.265384	0.967936	0.941631

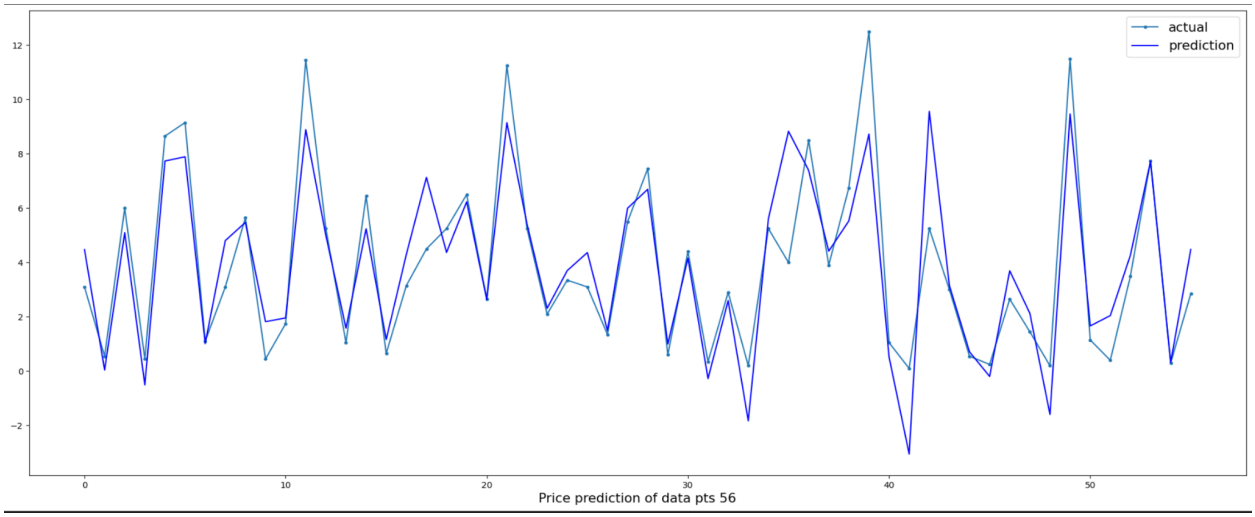
- The predicted values of all the models are compared with the actual values.

```
predicted_values
✓ 0.0s
```

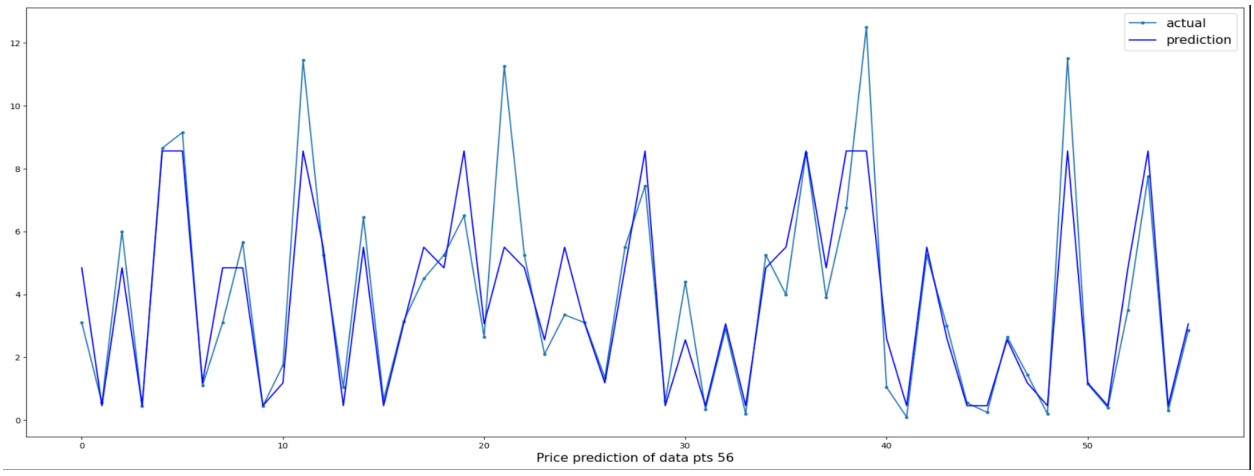
	y_test	MultipleLinearRegressor	DecisionTreeRegressor	RandomForestRegressor	SupportVectorRegressor	PolynomialRegressor	DecisionTreeAfterPruning
0	3.10	4.474857	4.844565	3.8760	3.458577	2.915143	3.890000
1	0.55	0.041632	0.461333	0.5585	0.904295	0.705167	0.593871
2	6.00	5.098738	4.844565	5.0100	4.595790	4.600303	4.642857
3	0.45	-0.504834	0.461333	0.5002	0.587295	0.432192	0.593871
4	8.65	7.733441	8.560294	8.9078	8.689321	8.655265	8.400000
5	9.15	7.891276	8.560294	10.4536	9.014991	9.320656	11.075000
6	1.10	1.015427	1.186190	1.1228	0.844062	1.022068	1.134737
7	3.10	4.804847	4.844565	3.9810	3.869998	3.509900	3.890000
8	5.65	5.476057	4.844565	5.8915	5.954001	6.029469	6.120000
9	0.45	1.822285	0.461333	0.4376	0.576390	0.385503	0.319655
10	1.75	1.960933	1.186190	1.4985	1.268610	1.689218	1.675000

- The predicted graphs of the predicted_y of all the models with the acutal_y are plotted.

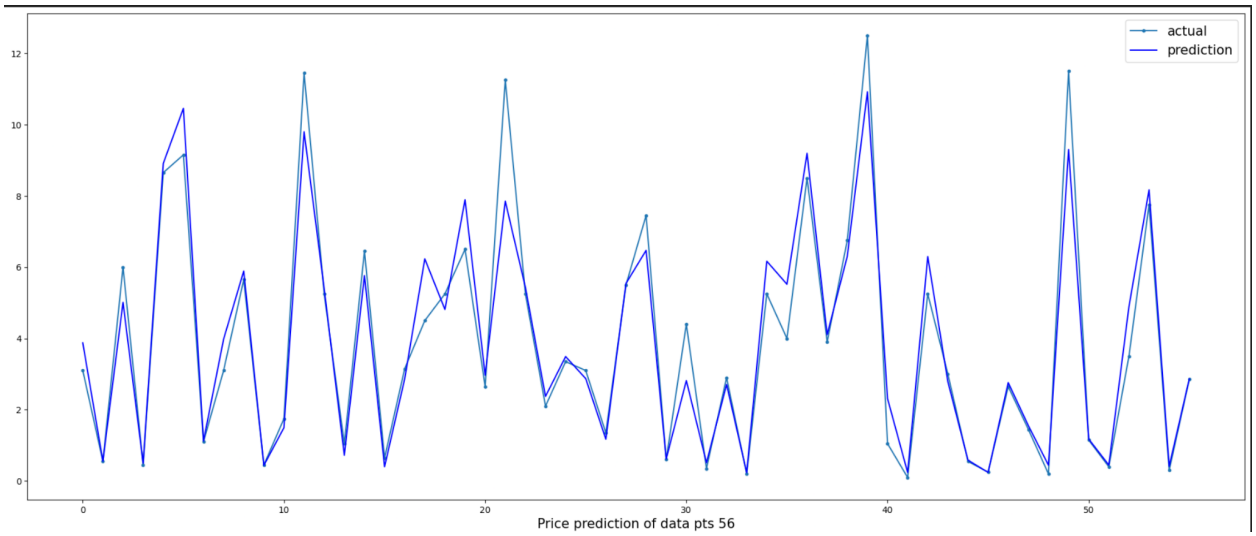
Multiple regression



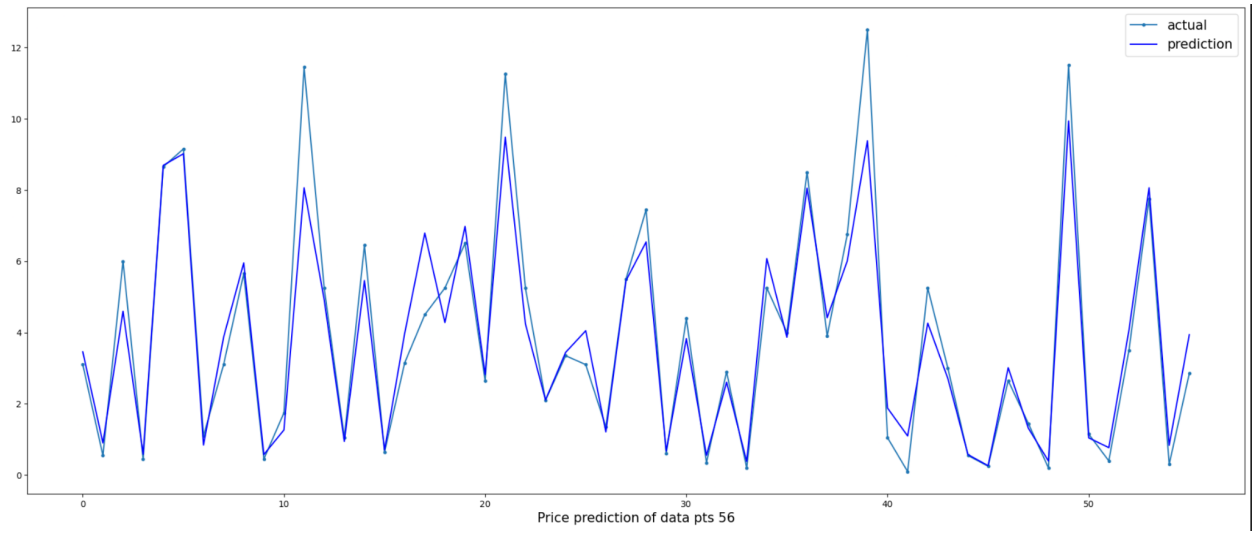
Decision Tree



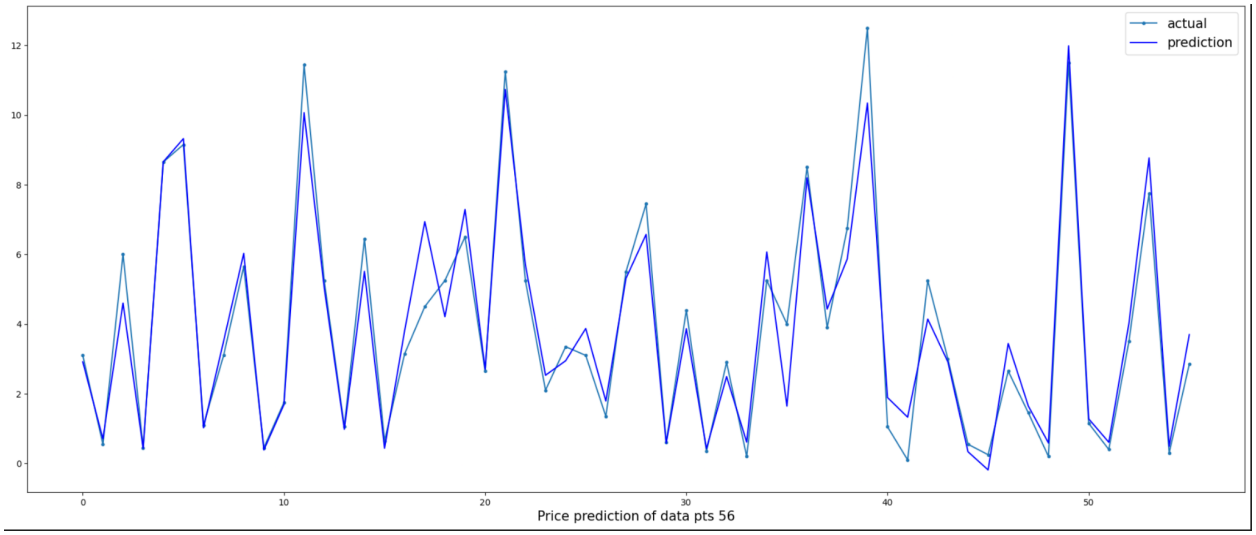
Random Forest



Support Vector Machine



Polynomial Regression



Conclusion

Models like Random Forest, Support Vector Machine and Polynomial regression give the best results for the dataset. Decision tree overfits the data by giving better accuracy for the training set and less with the testing set. But after Pre Pruning it gives relatively good results.