FLIP ROBO

Project Name: Customer Retention

Submitted by:

Dhrubajyoti Mandal

Acknowledgement

The success and final outcome of the machine learning requires a lot of guidance and assistance from some people and I am extremely privileged to have got this all among the completion of my course and few of the projects. All that I have done is only due to such supervision and assistance and I would not forget to thank them.

I respect and thank FLIP ROBO Technologies, for providing me this opportunity to do the carcerand project work and giving me all support and guidance, which made me complete the course.

I would like to thanks my mentor, Sapna Verma who guided me at every point of the project.

Introduction to Problem

CAR PRICE PREDICTION

With the covid 19 impact in the market, we have seen lot of changes in the car market. Now some cars are in demand hence making them costly and some are not in demand hence cheaper. One of our clients works with small traders, who sell used cars. With the change in market due to covid 19 impact, our client is facing problems with their previous car price valuation machine learning models. So, they are looking for new machine learning models from new data. We have to make car price valuation model. This project contains two phase Data Collection Phase You have to scrape at least 5000 used cars data. You can scrape more data as well, it's up to you. more the data better the model

In this section You need to scrape the data of used cars from websites (Olx, cardekho, Cars24 etc.) You need web scraping for this. You have to fetch data for different locations. The number of columns for data doesn't have limit, it's up to you and your creativity. Generally, these columns are Brand, model, variant, manufacturing year, driven kilometres, fuel, number of owners, location and at last target variable Price of the car. This data is to give you a hint about important variables in used car model. You can make changes to it, you can add or you can remove some columns, it completely depends on the website from which you are fetching the data. Try to include all types of cars in your data for example- SUV, Sedans, Coupe, minivan, Hatchback.

Note – The data which you are collecting is important to us. Kindly don't share it on any public platforms. Model Building Phase

After collecting the data, you need to build a machine learning model. Before model building do all data pre-processing steps. Try different models with different hyper parameters and select the best model.

Motivation for the Problem Undertaken

"Actually, car selling and buying has been a tiresome way over few years but the journey has been smoothening through-Cars24 i.e., hassle-free, safe, worth pricing and proper documentation."

Owing to the COVID pandemic the transit systems of India have been shut off completely as the country resorted to a nationwide lockdown. The present times have observed an increasing number of people leaning towards buying used cars over purchasing new ones. Since the pandemic has had dire implications upon the economy, purchasing new cars has become an option which could be avoided, and used cars have slowly become more preferable.

Analytical Problem Framing

- Dataset Representation

```
df=pd.read_csv(r'Cars24.csv')
df.head()
```

	Unnamed: 0	Brand	model	variant	Transmission	Year of Manufacturing	Driven_in_km	Fuel_type	Number of owner	Location	Price
0	0	Elite	Hyundai	ASTA 1.4 CRDI	Manual	2016	43221	Diesel	2	Bengaluru	686299
1	1	Baleno	Maruti	DELTA 1.2 K12	Manual	2016	13400	Petrol	1	Bengaluru	625599
2	2	Swift	Maruti	ZDI AMT	Automatic	2018	78534	Diesel	1	Bengaluru	746099
3	3	Verna	Hyundai	1.6 SX VTVT	Manual	2017	29525	Petrol	1	Bengaluru	903199
4	4	Ameo	Volkswagen	HIGHLINE 1.5	Manual	2016	56225	Diesel	1	Bengaluru	618999

```
df.shape
```

```
(4533, 11)
```

Cosine summary of the dataset

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4533 entries, 0 to 4532
Data columns (total 11 columns):
 #   Column                 Non-Null Count  Dtype 
---  ------                 --------------  ----- 
 0   Unnamed: 0             4533 non-null   int64 
 1   Brand                  4533 non-null   object
 2   model                  4533 non-null   object
 3   variant                4533 non-null   object
 4   Transmission           4533 non-null   object
 5   Year of Manufacturing  4533 non-null   int64 
 6   Driven_in_km           4533 non-null   int64 
 7   Fuel_type              4533 non-null   object
 8   Number of owner        4533 non-null   int64 
 9   Location               4533 non-null   object
 10  Price                  4533 non-null   int64 
dtypes: int64(5), object(6)
memory usage: 389.7+ KB
```

Statistical Summary of the dataset

```
df.describe().T
```

	count	mean	std	min	25%	50%	75%	max
Unnamed: 0	4533.0	2266.000000	1308.708715	0.0	1133.0	2266.0	3399.0	4532.0
Year of Manufacturing	4533.0	2016.578866	2.238881	2007.0	2015.0	2017.0	2018.0	2021.0
Driven_in_km	4533.0	45660.038165	32968.244574	58.0	22192.0	38846.0	61499.0	280921.0
Number of owner	4533.0	1.234061	0.480112	1.0	1.0	1.0	1.0	4.0
Price	4533.0	607334.737922	311146.446567	120399.0	405499.0	529399.0	700299.0	3506099.0

Observation:

- 10 Independent variables with Price as target variables.
- From the dataset we can infer that it is clearly a regression problem.
- The dataset consists of 11 rows and 4533 columns.
- Statistical summary of the dataset is okay.

Data Sources and their formats & inferences

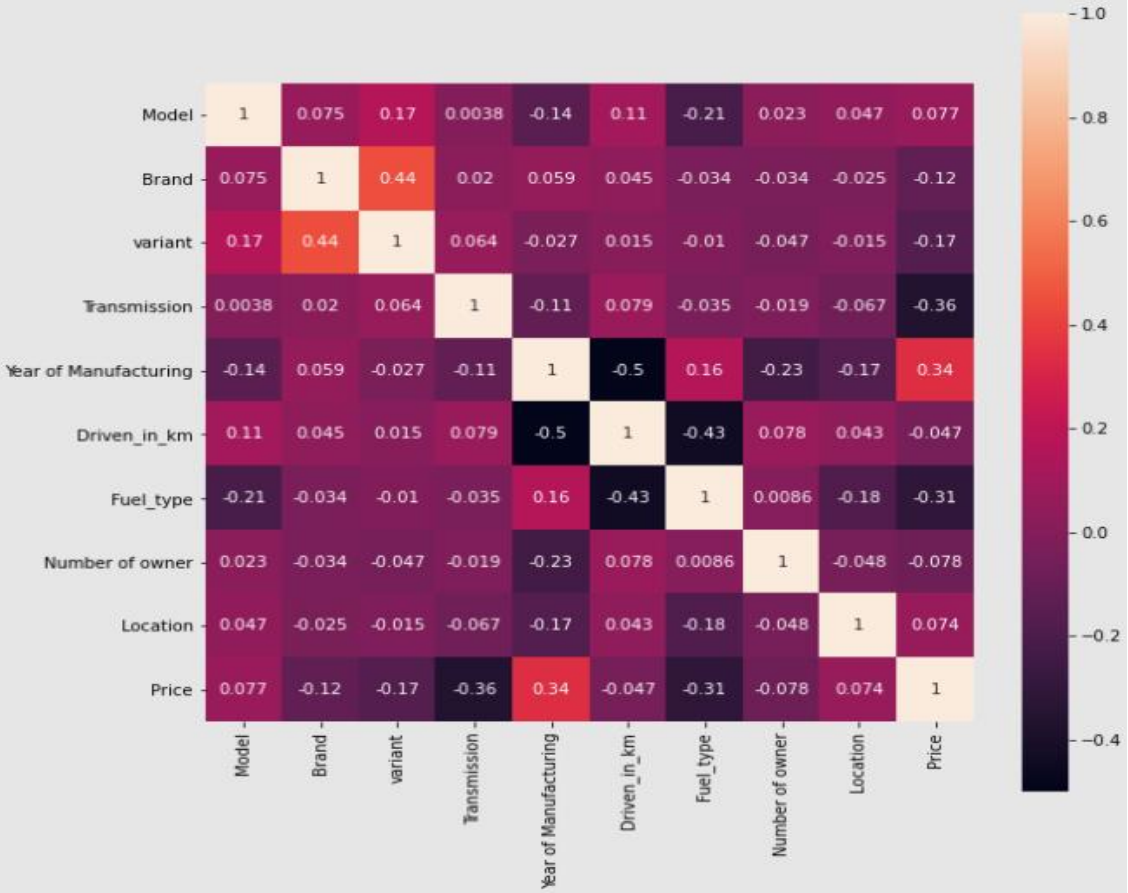
- Brand: Brand of the car.
- Model: Model name of the car
- variant: Variant specifies the type of the engine, model type (base model, top model etc..).
- Transmission: It specifies if it is an automatic or manually driven gear car.
- Year of Manufacturing: It gives the year manufacturing year of the car i.e., with which we can find how old the car is.
- Driven_in_km: Identifies the total distance driven by the car in km.
- Fuel_type: By this feature we can know the type of fuel used for the car(petrol/diesel/CNG)
- No_of_owner: it says how many times the car is being resell i.e., change in number of owners.
- Location: Specifies the location.
- Price: The price of the car

10 Independent variables with Price as target variables

- CARS24 is primarily a business for resale of used cars. The platform's business model is based on the criteria's of purchasing a car from their owner at the most suitable price and competent price in comparison with the other alternate services for car resale around the neighbourhood. The platform is basically focused on offering the Indian consumers a more suitable and easier alternative as opposed to various other banal approaches of selling used cars, in order to help make the whole procedure smoother and more straightforward.
- The platform facilitates an assured price for every car, irrespective of its condition, age as well as model. Cars24 also allows the users to obtain a sketchy valuation, on the basis of the information provided of their car online on the platform. The users who sell their cars through the company's platform obtain the payment instantly.

Data Inputs- Logic- Output Relationships

- Correlation –



Observation:

Brand Name, Variant, Manufacturing year and Price are showing mild correlation.

Assumption for the problem:

From the dataset we can infer that it is clearly a regression problem.

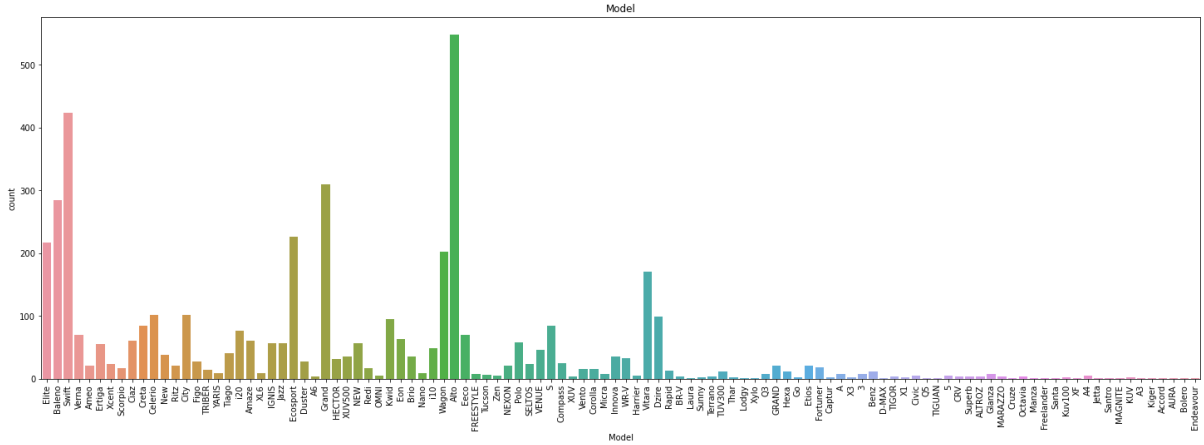
Hardware and Software Requirements and Tools Used

- Jupyter Notebook
- Ms-Paint
- MS-PowerPoint
- MS-Word
- Laptop
- Good internet connectivity

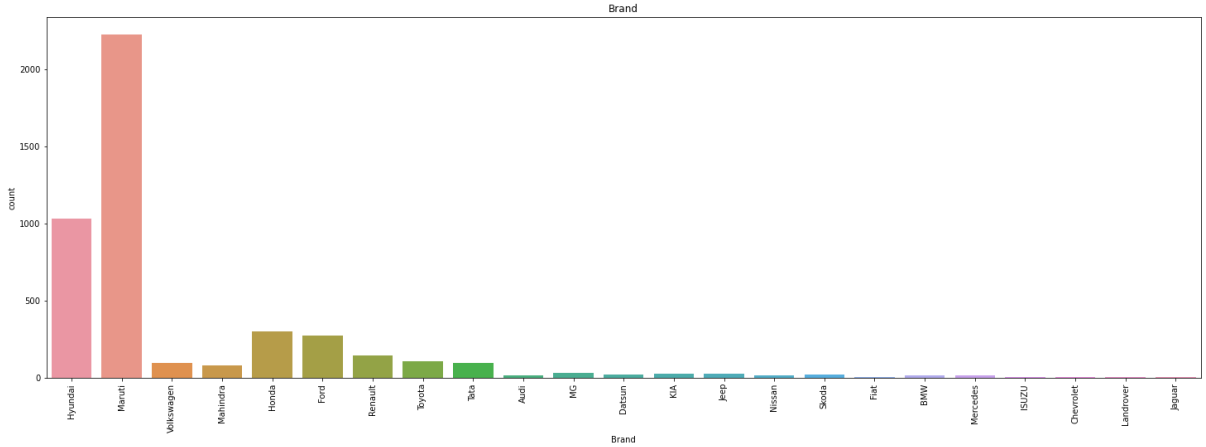
Visualization

Count plots of categorical features –

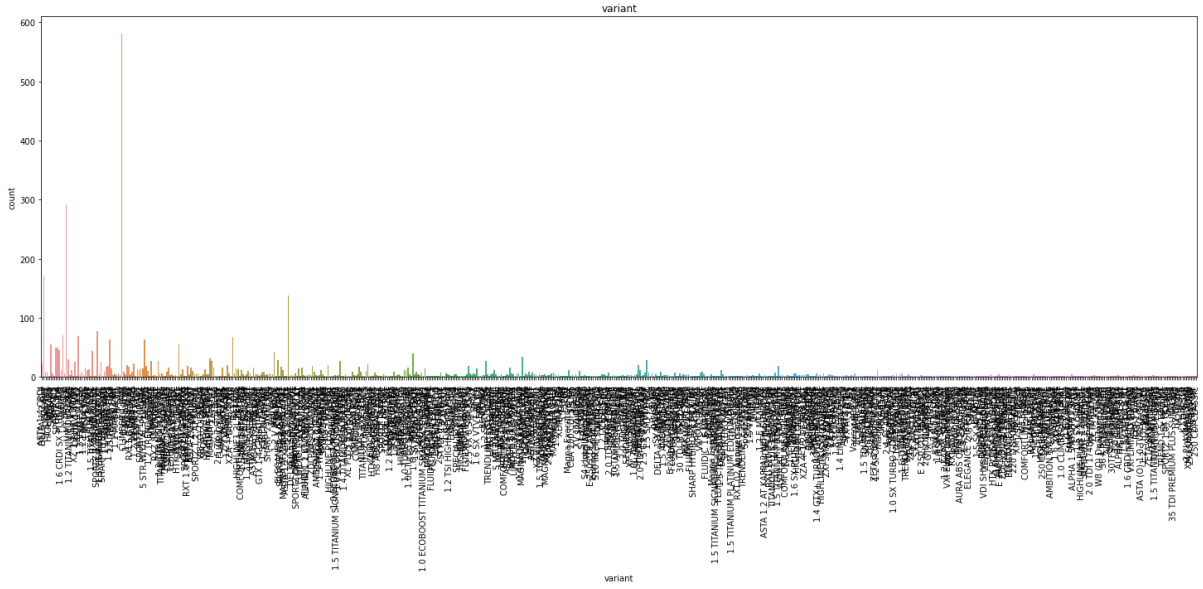
1. Model



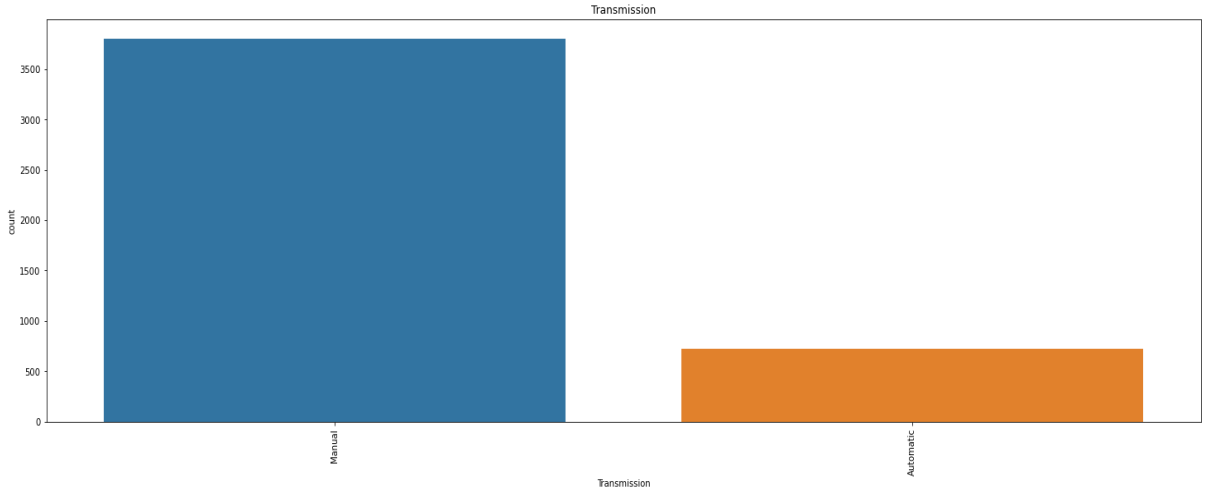
2. Brand



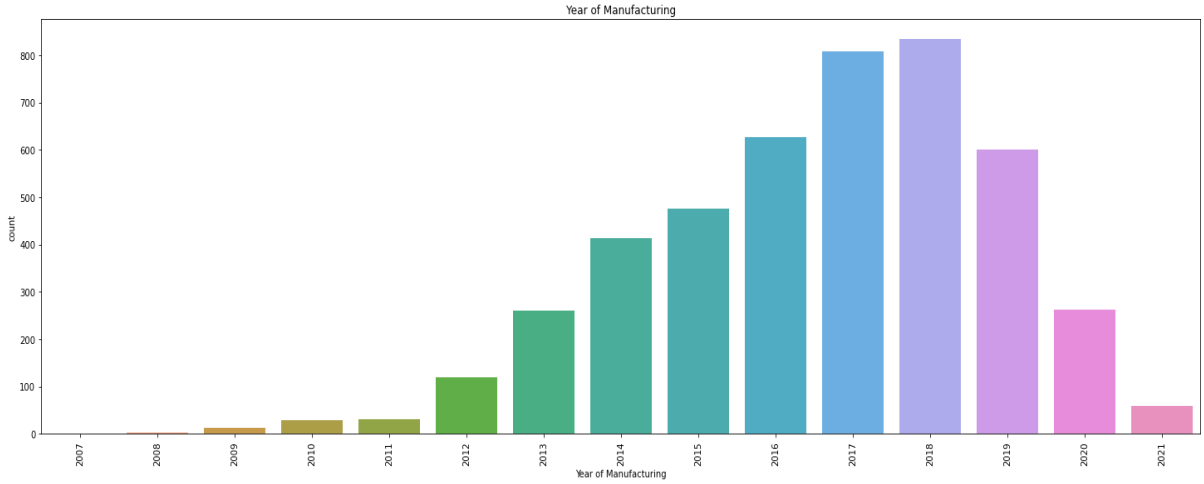
3. Variant



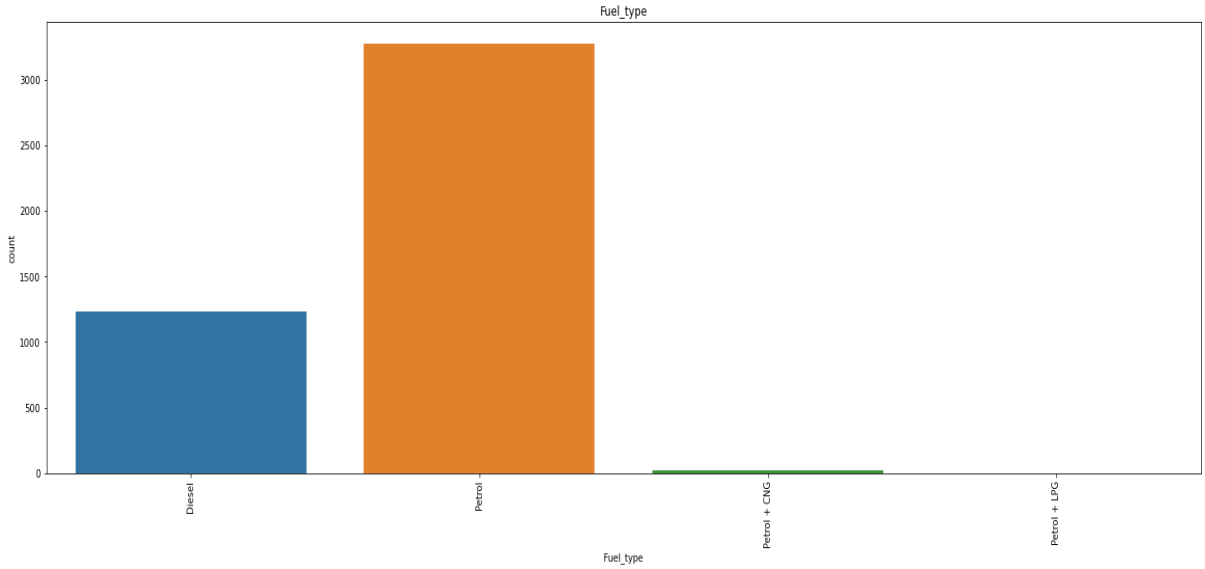
4. Transmission



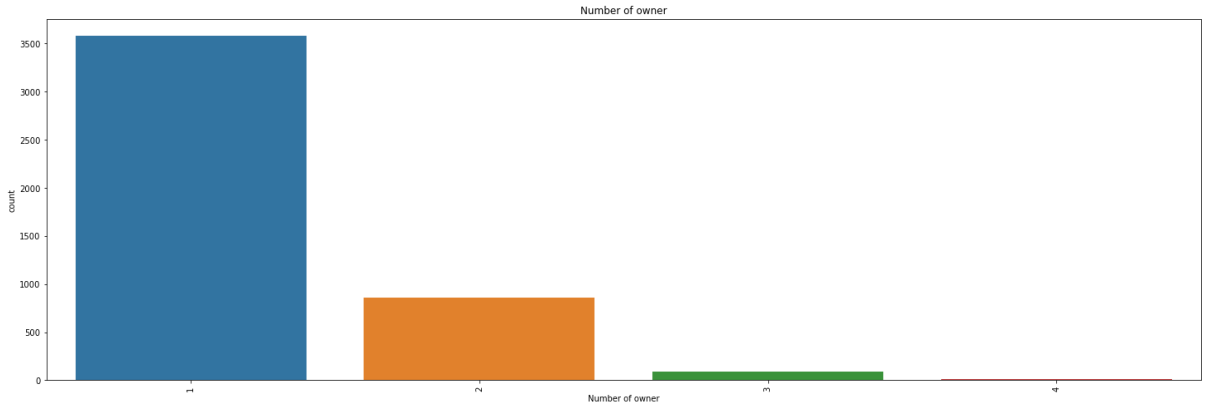
5. Year of Manufacturing



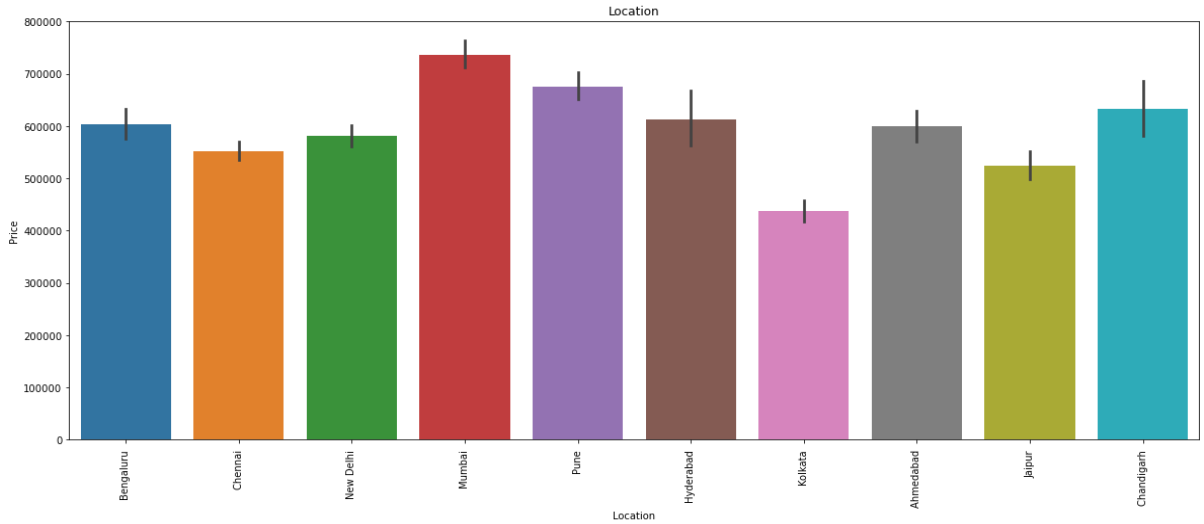
6. Fuel Type



6. Number of owners

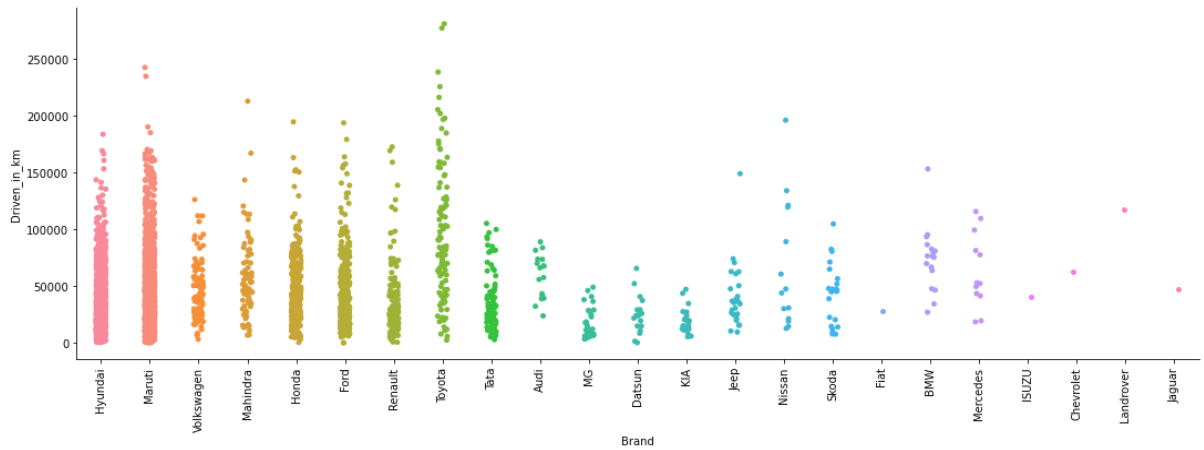


6. Location

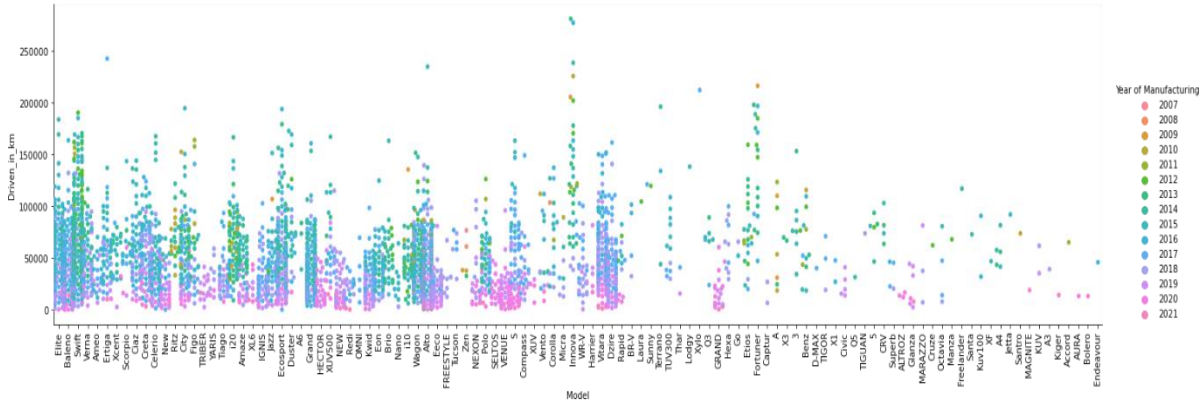


The next two catplot show how much kms different vehicles of different brands had driven.

1. Brand vs Driven_in_km



2. Model vs Driven_in_km



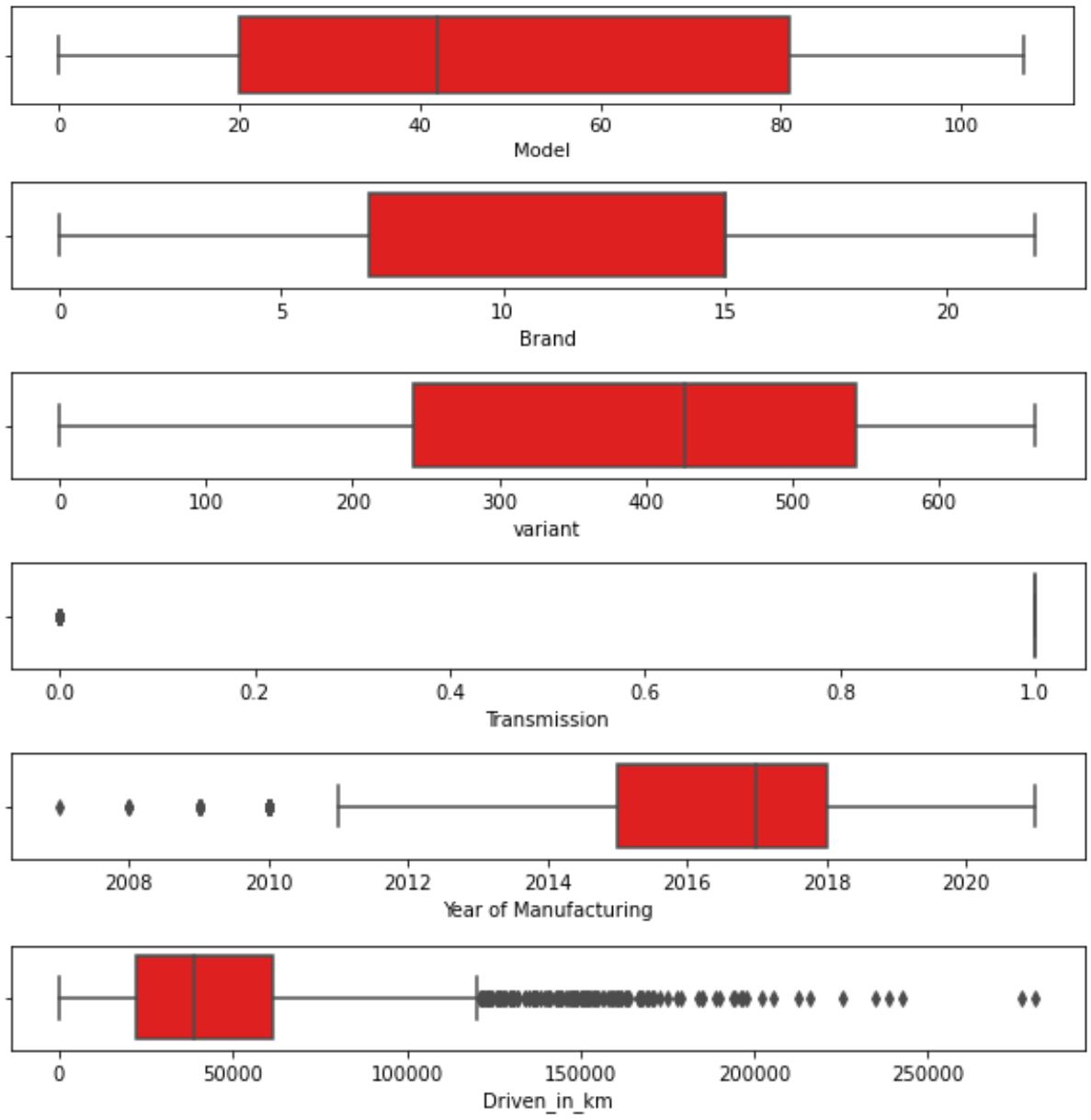
Observations from all the above graphs:

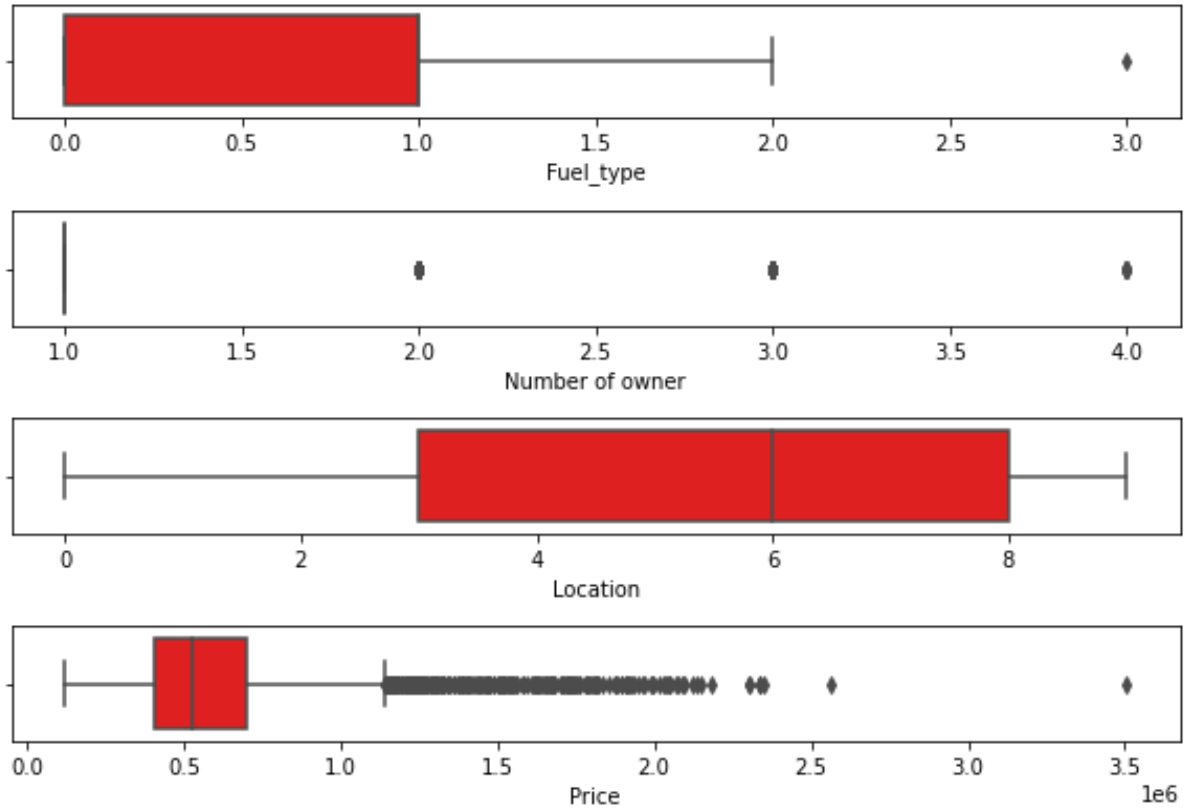
- Largest sale is Alto car, followed by swift then elite and so on as per the visualization.
- Highest sale is of Maruti Brand followed by Hyundai and Honda.
- Mostly manual cars are on sale.
- No of petrol are maximum on sale.
- People staying art Mumbai and Pune are selling their cars mostly followed by New Delhi location.
- We can infer that VXI and LXI model are maximum in number for sale.
- Most of the cars are sold during the year 2017 and 2018.
- Most of the cars that are for resale are driven for the less than 50000 km.3. Most of the cars on Cars24 were on sell for first time.
- Also, no of owners and Driven km are skewed.
- But we can see that our manufacturing year is skewed.
- Our target variable price feature is skewed.
- Benz having highest price followed by Superb, Harrier and so on.
- Mercedes brand is having highest price followed by Jaguar.
- Automatic cars are higher in price as compared to manual cars.

- Diesel cars are having higher resale value compared to petrol and combination of petrol and CNG.
- Price of cars at Pune and Mumbai are highest followed by Chandigarh.

Outlier Detection –

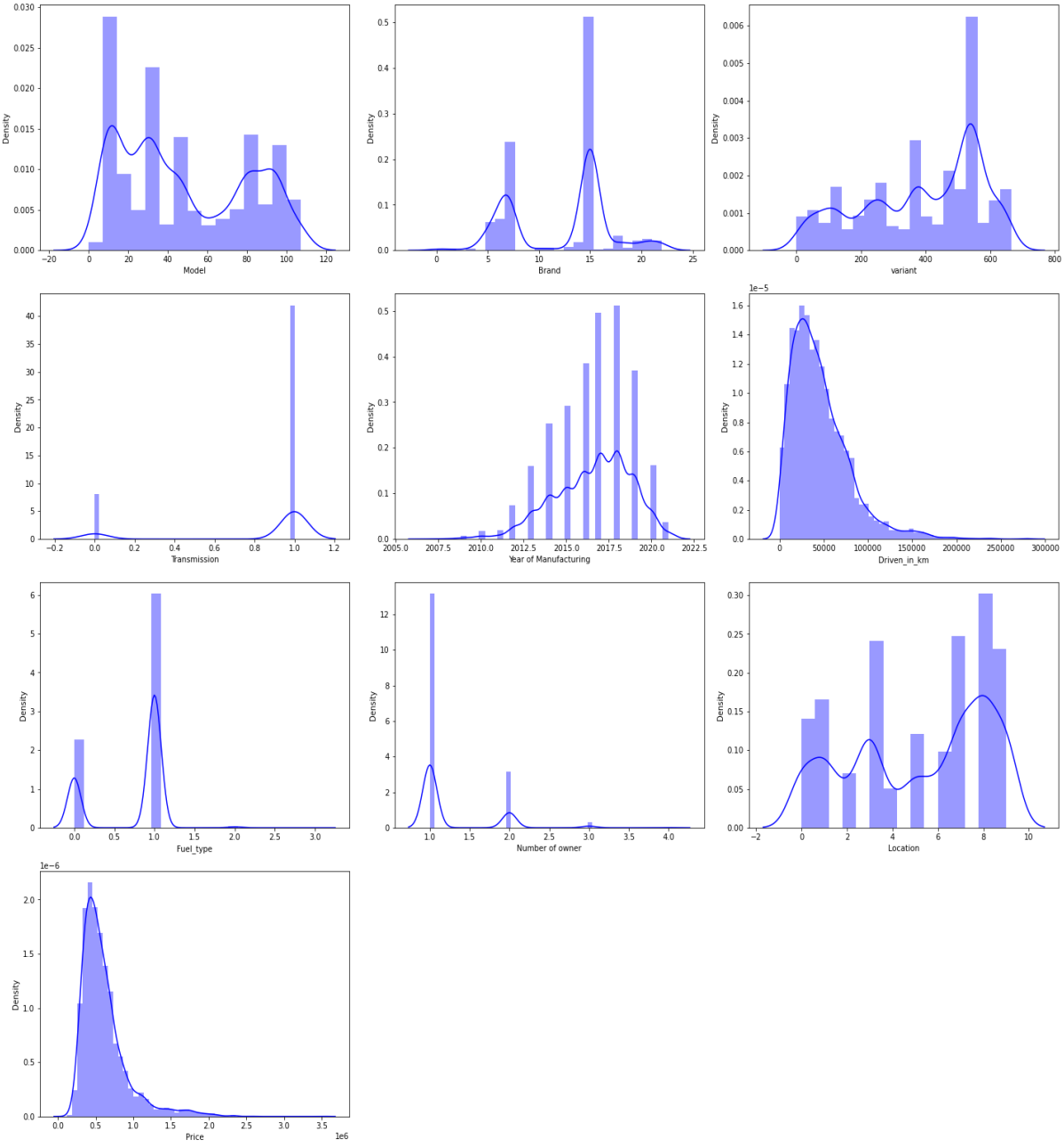
The below boxplots will show if there are outliers presents on the columns or not.





From the above graphs we have confirmed that there are some outliers present in the dataset.

Next, let's find out if there is any skewness in the dataset along with the distribution curve, i.e., either it is following the Gaussian theorem or not using distplot.



Observation:

- Not all the features are following Gaussian Distribution Theorem.
- There is skewness present in the dataset.

Next step of feature engineering is scaling, so that Skewness and the Outliers will be removed but before that let's divide the dataset into features and label.

Split the dataset into features and target variables

```
# Independent Variables
X=df.drop('Price', axis=1)
# Dependent Variables
y=df['Price']
```

- X = features
- y = target

Scaling the features

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()

features = scaler.fit_transform(X)
```

Above I have scaled the features so that I can treat outliers as well as skewness at the same time

Model Building

In model building process,

I have used four different types of algorithms.

- K-Neighbors Regressor
- Random Forest Regressor
- Decision Tree Regressor
- Gradient Boosting Regressor

```
# K-Neighbors Regressor
from sklearn.neighbors import KNeighborsRegressor
knr = KNeighborsRegressor()
beststate(knr)
```

```
Best Random State      :  74
Best R2_Score          :  0.6655658259372805
Cross Validation Score :  0.5257732021800743

Time taken by model for prediction 0.4502 seconds
```

```
# Random Forest Regressor
from sklearn.ensemble import RandomForestRegressor
rf = RandomForestRegressor()
beststate(rf)
```

```
Best Random State      :  74
Best R2_Score          :  0.887996254907386
Cross Validation Score :  0.8366825111439822

Time taken by model for prediction 14.3518 seconds
```

```
# Decision Tree Regressor
from sklearn.tree import DecisionTreeRegressor
dt = DecisionTreeRegressor()
beststate(dt)
```

```
Best Random State      :  74
Best R2_Score          :  0.8553598917496278
Cross Validation Score :  0.6733960733595582

Time taken by model for prediction 0.2330 seconds
```

```
# Gradient Boosting Regressor
from sklearn.ensemble import GradientBoostingRegressor
gbr = GradientBoostingRegressor()
beststate(gbr)
```

```
Best Random State      :  73
Best R2_Score          :  0.8351503982424882
Cross Validation Score :  0.7764658229205363

Time taken by model for prediction 4.9132 seconds
```

On applying different algorithms, we can clearly find that both decision tree and random forest classifiers are providing good score. Now, let's tune the parameters of this two algorithms to find the best model.

HYPER TUNING THE MODEL

1. Hyper parameter tuning of the Gradient Boosting Regressor using GridSearchCV

```
# Hyper Parameter Tuning with Gradient Boosting Regressor

X_train, X_test, y_train, y_test = train_test_split(features, y, test_size=0.20, random_state=72)

from sklearn.model_selection import GridSearchCV

param_grid = {"min_samples_leaf" : [1,2,3],
              "min_samples_split" : [2,3,4],
              "n_estimators" : [100,200],
              "learning_rate" : [0.1,0.2]}
grid_search = GridSearchCV(gbr, param_grid=param_grid)
grid_search.fit(X_train, y_train)
grid_search.best_params_
```

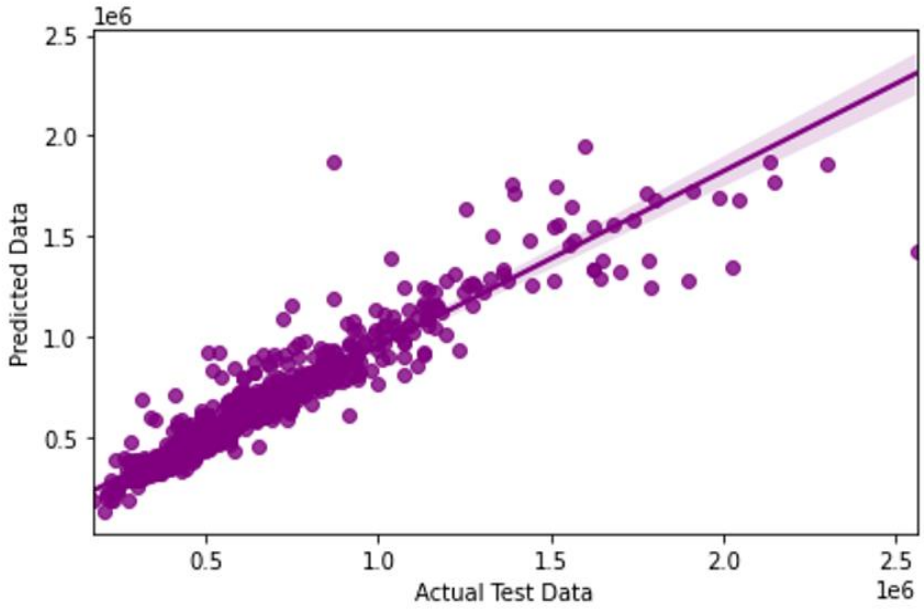
```
{'learning_rate': 0.2,
 'min_samples_leaf': 2,
 'min_samples_split': 3,
 'n_estimators': 200}
```

```
# Final Model
best_model = GradientBoostingRegressor(learning_rate=0.2,min_samples_split=2,min_samples_leaf=2,n_estimators=200)
X_train, X_test, y_train, y_test = train_test_split(features, y, test_size=0.20, random_state=72)
best_model.fit(X_train, y_train)
y_pred = best_model.predict(X_test)
print("r2_score :",r2_score(y_test, y_pred))

sns.regplot(y_test,y_pred, color='purple')
plt.xlabel("Actual Test Data")
plt.ylabel("Predicted Data")
plt.tight_layout()
```

```
r2_score : 0.8883970013368551
```

REGPLOT TO VISUALIZE ACTUAL TEST DATA VS PREDICTED DATA AND ALSO FIND THE BEST FIT LINE OF GRADIENT BOOSTING REGRESSOR.



2. Hyper parameter tuning of the Gradient Boosting Regressor using GridSearchCV

```
# Hyper Parameter Tuning with Randomized SearchCV  (Random Forest Regressor)

X_train, X_test, y_train, y_test = train_test_split(features, y, test_size=0.20, random_state=72)

from sklearn.model_selection import RandomizedSearchCV
rfg = RandomForestRegressor()

params = {'n_estimators':[13,15, 100, 200],
          'max_depth':[10,15, 20, 25],
          'min_samples_split':[5,6,7],
          'min_samples_leaf':[8,9,10]
          }

grid_search = RandomizedSearchCV(rfg, param_distributions=params)
grid_search.fit(X_train, y_train)
grid_search.best_params_
```

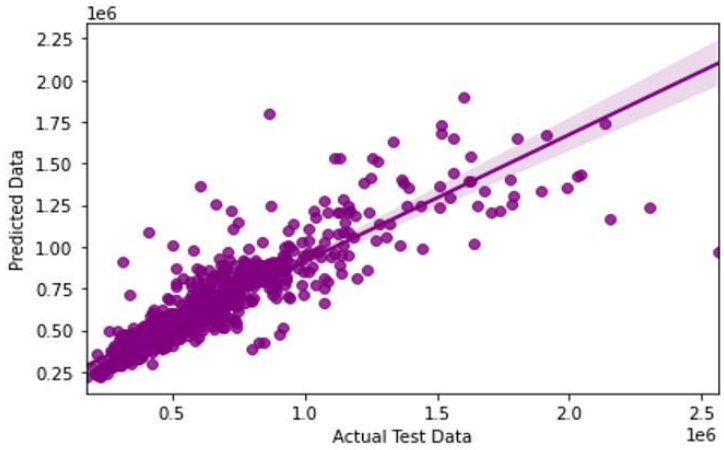
```
{'n_estimators': 15,
 'min_samples_split': 5,
 'min_samples_leaf': 8,
 'max_depth': 10}
```

```
# Final Model
best_model_ = RandomForestRegressor(min_samples_split=5,min_samples_leaf=8,n_estimators=15, max_depth=10)
X_train, X_test, y_train, y_test = train_test_split(features, y, test_size=0.20, random_state=72)
best_model_.fit(X_train, y_train)
y_pred_ = best_model_.predict(X_test)
print("r2_score :",r2_score(y_test, y_pred_))

sns.regplot(y_test,y_pred_, color='purple')
plt.xlabel("Actual Test Data")
plt.ylabel("Predicted Data")
plt.tight_layout()
```

```
r2_score : 0.7741258782085034
```

REGPLOT TO VISUALIZE ACTUAL TEST DATA VS PREDICTED DATA AND ALSO FIND THE BEST FIT LINE OF RANDOM FOREST REGRESSOR



Hence, after comparing both the model, we found that Gradient Boost model is the best model as it gives higher r2 score and the cost function is also less in this case.

Interpretation of the Results

- As we can see that after hypertuning the model, Gradient Boost is giving the best result.
- This model may help people solve real life problems, before CARS24 came into existence, selling a car in 1 visit and getting the best price for it was merely a dream.
- Even as the COVID-19 restrictions ease, CARS24 is seeing a good number of seller and buyer to the pre-COVID numbers hence there is a good turn around in the market.

CONCLUSION

- The increased prices of new cars and the financial incapability of the customers to buy them, Used Car sales are on a global increase.
- Therefore, there is an urgent need for a Used Car Price Prediction system which effectively determines the worthiness of the car using a variety of features.
- The proposed system will help to determine the accurate price of used car price prediction.

FINAL OBSERVATION:

I found that GradientBoostingRegressor is giving the best r2_score of 0.8883970013368551 (88.9%) after tuning.

Hence we can carry on with the model for prediction.

Sample of prediction:

	Y Test	Pred
2330	1165399	1150704.56
2965	528899	524699.80
3351	253599	276280.58
1489	365999	383017.29
648	942799	785016.23

Limitations of this work and Scope for Future Work

- In future this machine learning model may bind with various website which can provide real time data for price prediction.
- Also, we may add large historical data of car price which can help to improve accuracy of the machine learning model.
- We can build an android app as user interface for interacting with user. For better performance, we plan to judiciously design deep learning network structures, use adaptive learning rates and train on clusters of data rather than the whole dataset.
- This allows owners of used cars to be able to sell them in a single visit at any branch of the platform.

****

Thank you