Used Car Price Prediction

Submitted by:

Maheshkumar

Ota

ACKNOWLEDGMENT

I express my sincere gratitude to Flip Robo Technologies for giving me the opportunity to work on this project on Used Car Price Prediction using machine learning algorithms. I acknowledge my indebtedness to the authors of online articles titled: “Factors which affect used car valuation price” and “Just What Factors Into The Value Of Your Used Car?” for providing me with invaluable insights and knowledge of the various factors that determine the price of a used car.

INTRODUCTION

Business Problem Framing

With the covid 19 impact in the market, there have been 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. With the change in market due to covid 19 impact, our client there is a need for new machine learning models from new data. Therefore, new car price valuation model is required to be made.

Conceptual Background of the Domain Problem

Predictive modelling, Regression algorithms are some of the machine learning techniques used for predicting Used Car prices. Identifying various relevant features of a car, its present condition, ownership and total usage by previous owner(s) are crucial for working on the project as they determine its valuation.

Review of Literature

3 Online Articles, namely: “Factors which affect used car valuation price” and “Just What Factors Into The Value Of Your Used Car?” were reviewed and studied to gain insights into all the attributes that contribute to the price of a used car.

It is learnt that Economic Factors, Vehicle Make, Vehicle Class and Body Style, Mileage, Transmission Type and technology are some the most important factors that determine a used car’s valuation.

- https://www.moneycrashers.com/factors-affect-used-cars-resale-value/
- https://www.investopedia.com/articles/investing/090314/just-what-factors-value-your-used-car.asp
- https://autoportal.com/articles/factors-which-affect-used-car-valuation-price-6446.html

Motivation for the Problem Undertaken

With the covid 19 impact in the market, we have seen lot of changes in the car market. Now there is a boom in demand for cars in the market, hence making them costly, while some are not in demand hence cheaper. There is a need for building machine learning models from new data that would help accurately predict the valuation of used cars based on various attributes.

Analytical Problem Framing

Mathematical/ Analytical Modeling of the Problem

Various Regression analysis techniques were used to build predictive models to understand the relationships that exist between used car price and features and attributes of the car. The Regression analysis models were used to predict the car price value for changes in car attributes. Regression modelling techniques were used in this Problem since Car Price data distribution is continuous in nature.

In order to forecast car price, predictive models such as ridge regression Model, Random Forest Regression model, Decision tree Regression Model, Support Vector Machine Regression model, Extreme Gradient Boost Regression and K Nearest Neighbours model were used to describe how the values of Car Price depended on the independent variables of various Car attributes.

Data Sources and their formats

The Dataset was compiled by scraping Data for various Car attributes and Price from https://droom.in/.

The data was converted into a Pandas Dataframe under various Feature and Label columns and saved as a .csv file.

```
DF.head(50)
```

	Unnamed: 0	Brand Name	Model Name	Year	Type	Total Kilometers Driven	Fuel Type	Owner	Location	Transmission	Seating Capacity	Mileage	Price
0	0	Maruti Suzuki Swift	Maruti Suzuki Swift VXi 2008	2008	Hatchback	99500	Petrol	Second Owner	munger	Manual	5	20.4	180000
1	1	Mahindra Marazzo	Mahindra Marazzo M8 8 STR 2019	2019	MUV	33000	Diesel	First Owner	kanchipuram	Manual	8	17.6	1335000
2	2	Honda City	Honda City 1.5 S MT 2012	2012	Sedan	77000	Petrol	First Owner	delhi	Manual	5	17.8	350000
3	3	Skoda Rapid	Skoda Rapid Ambition 1.6 TDI CR MT Plus Alloy ...	2013	Sedan	58000	Diesel	First Owner	bangalore	Manual	5	20.5	480000
4	4	Toyota Etios Liva	Toyota Etios Liva G 2014	2014	5 Seater	26700	Hatchback	MH 02	First Owner	17.71 kmpl	Manual	-	365000
5	5	Nissan Terrano	Nissan Terrano XL (P) 2018	2018	SUV	68000	Petrol	First Owner	ahmedabad	Manual	5	13.24	654050
6	6	Toyota Innova	Toyota Innova 2.5 GX 7 STR BS IV 2015	2015	MUV	92000	Diesel	First Owner	delhi	Manual	5	-	850000
7	7	Volkswagen Passat	Volkswagen Passat Highline DSG 2008	2008	Sedan	97000	Diesel	Third Owner	bangalore	Automatic	5	18.78	415000

Dataset Description

The Independent Feature columns are:

- Brand Name: Name of the Car Brand
- Model Name: Name of the specific Car Model of a Brand
- Year: Year of Manufacture
- Type: Car Model Type
- Total Kilometers Driven: Total Kilometers,for which the car has been so far driven.
- Fuel Type: Type of fuel used
- Owner: The ordinal number of previous owner
- Location: Availability Location of Car

- Transmission: Power Transmission type of the car
- Seating Capacity: Total number of passengers that the car can accomodate
- Mileage: Mileage of the Car

Target / Label Column:

- Price: Selling Price of the Car

Data Preprocessing Done

- Rows containing incorrectly entered data were removed first.
- Data in columns: Seating Capacity,Mileage,Price,Total Kilometers Driven,Year were converted to int64 / float data type.
- There were rows containing a character: '-' as data. These entries were first converted into NaN values and then imputed with data using most frequently occurring value imputation and KNN imputation techniques.
- Duplicate data elements which had their starting letters in upper case and lower case were converted to data elements starting with uppercase letters.
- Columns: Unnamed: 0(just a series of numbers) was dropped since it doesn't contribute to building a good model for predicting the target variable values.

Feature Engineering:

- In order to better understand the relationships between Car price and Car attributes, 'Brand','Model' and 'Variant' columns were created based on data of existing columns: 'Brand Name' and 'Model Name'.
- Column 'Car Age' was created based on data from 'Year'.

Data Inputs- Logic- Output Relationships

- The Datasets consist mainly of Float and Object data type variables. The relationships between the independent variables and dependent variable were analysed.

Hardware and Software Requirements and Tools Used

Hardware Used:

- Processor: AMD Ryzen 9 5900HX(8 Cores 16 Logical Processors)
- Physical Memory: 16.0GB (3200MHz)
- GPU: Nvidia RTX 3060 (192 bits), 6GB DDR6 VRAM, 3840 CUDA cores.

Software Used:

- Windows 10 Operating System
- Anaconda Package and Environment Manager: Anaconda is a distribution of the Python and R programming languages for scientific computing, that aims to simplify package management and deployment. The distribution includes data-science packages suitable for Windows and provides a host of tools and environment for conducting Data Analytical and Scientific works. Anaconda provides all the necessary Python packages and libraries for Machine learning projects.
- Jupyter Notebook: The Jupyter Notebook is an open-source web application that allows data scientists to create and share documents that integrate live code, equations, computational output, visualizations, and other multimedia resources, along with explanatory text in a single document.
- Python3: It is open source, interpreted, high level language and provides great approach for object-oriented programming. It is one of the best languages used for Data Analytics And Data science projects/application. Python provides numerous libraries to deal with mathematics, statistics and scientific function.

- Python Libraries used:
    - Pandas: For carrying out Data Analysis, Data Manipulation, Data Cleaning etc
    - Numpy: For performing a variety of operations on the datasets.
    - matplotlib.pyplot, Seaborn: For visualizing Data and various relationships between Feature and Label Columns
    - Scipy: For performing operations on the datasets
    - Statsmodels: For performing statistical analysis
- sklearn for Modelling Machine learning algorithms, Data Encoding, Evaluation metrics, Data Transformation,Data Scaling, Component analysis,Feature selection etc.

Exploratory Data Analysis

Visualizations

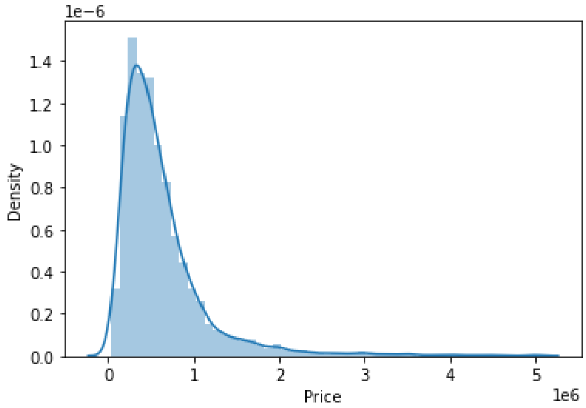
Barplots,Distplots,Boxplots,Countplots,lineplots were used to

visualise the data of all the columns and their relationships

with Target variable.

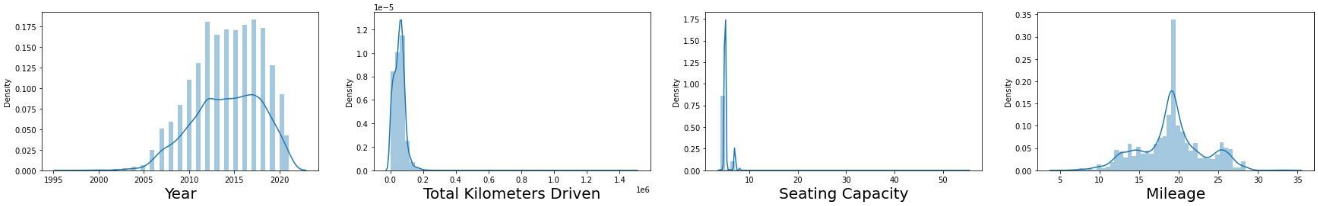
Univariate Analysis

Analyzing the Target Class



From the graph above it is observed that the Price data forms a continuous distribution and the distribution is skewed.

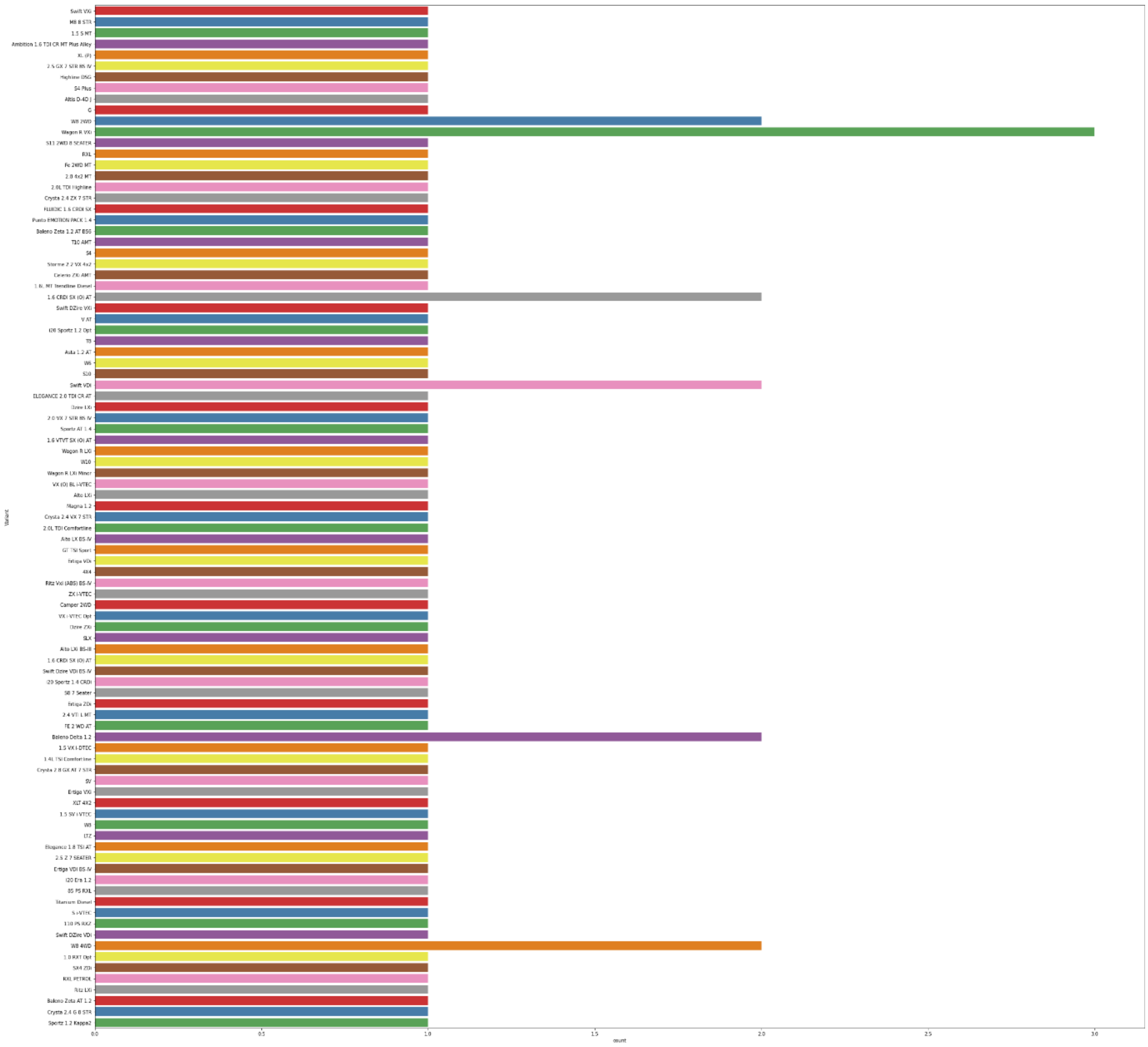
Analyzing the Feature Columns



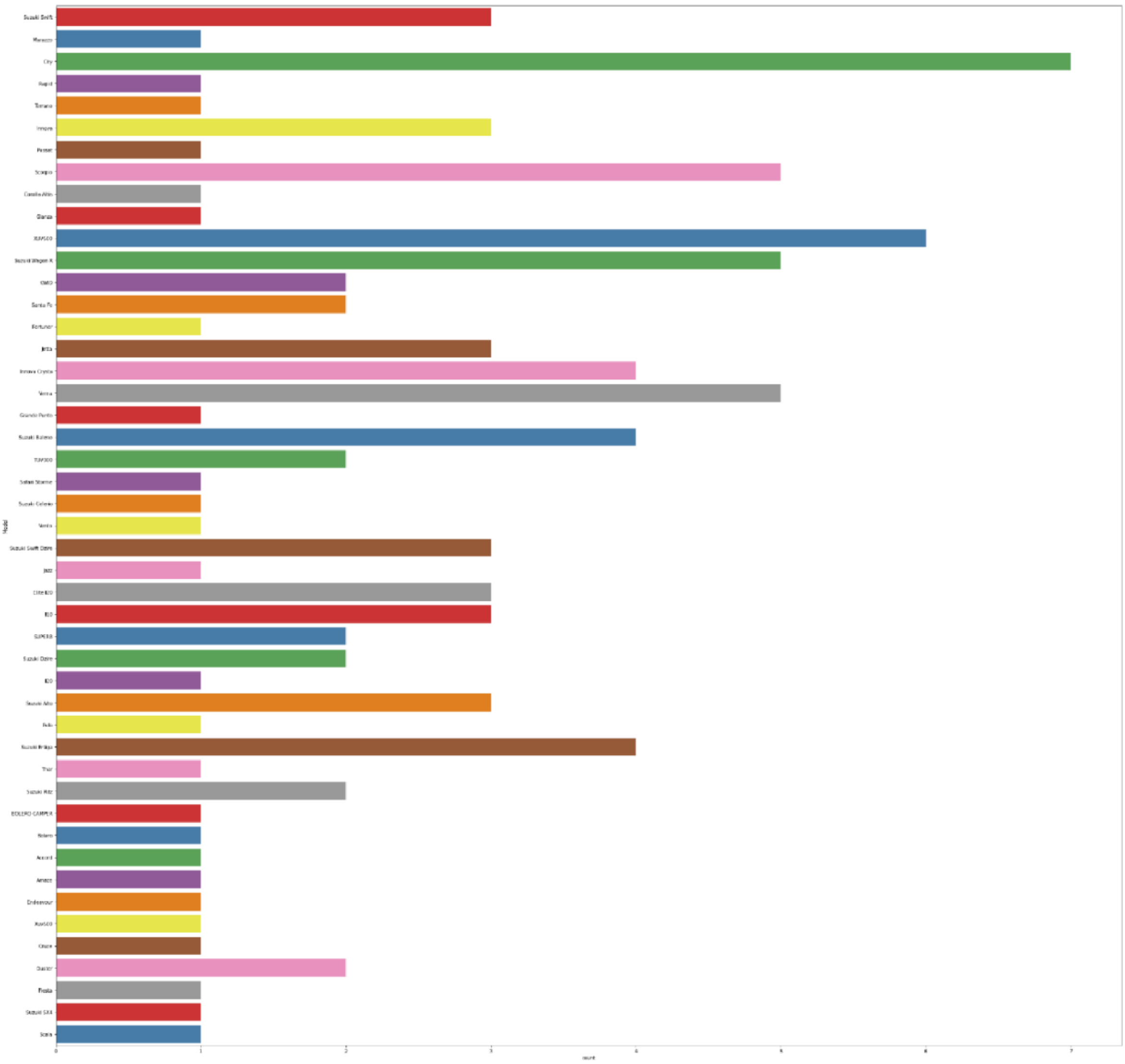

```
1  DF[DF.columns[DF.dtypes != 'object']].skew()
```

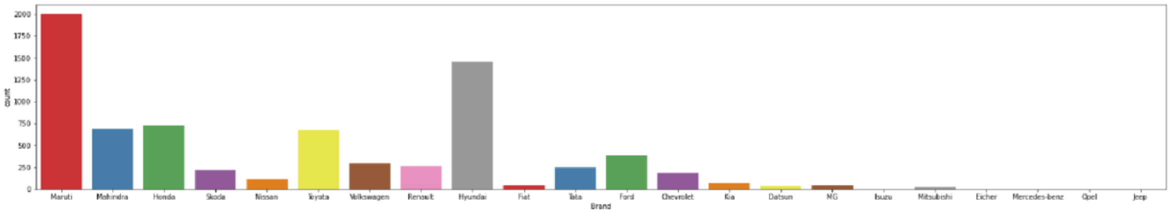
```
Year                      -0.339892
Total Kilometers Driven   11.284507
Seating Capacity          24.105809
Mileage                   -0.008898
Price                      3.136393
dtype: float64
```

Year and Mileage columns look normally distributed, while Total kilometers Driven and Seating Capacity are skewed

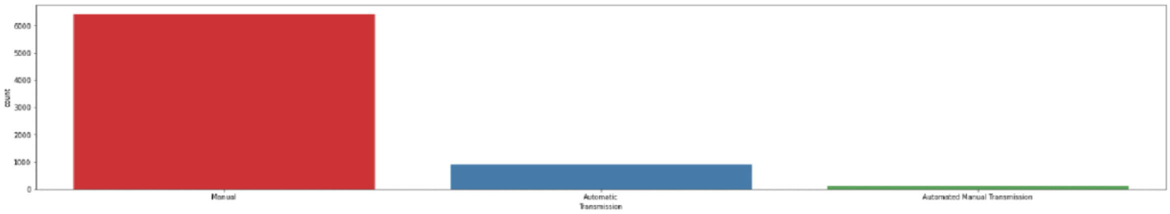


Swift VXI
M8 8 STR
1.5 S MT
Ambition 1.6 TDI CR MT Plus Alloy
XL (P)
2.5 GX 7 STR BS IV
Highline DSG
S4 Plus
Altis D-4D J
G
W8 2WD
Wagon R VXI
S11 2WD 8 SEATER
RXL
Fe 2WD MT
2.8 4x2 MT
2.0L TDI Highline
Crysta 2.4 ZX 7 STR
FLUIDIC 1.6 CRDI SX
Punto EMOTION PACK 1.4
Baleno Zeta 1.2 AT BS6
T10 AMT
S4
Storme 2.2 VX 4x2
Celerio ZXi AMT
1.6L MT Trendline Diesel
1.6 CRDI SX (O) AT
Swift DZire VXi
V AT
i20 Sportz 1.2 Opt
T8
Asta 1.2 AT
W6
S10
Swift VDI
ELEGANCE 2.0 TDI CR AT
Dzire LXi
2.0 VX 7 STR BS IV
Sportz AT 1.4
1.6 VTVT SX (O) AT
Wagon R LXi
W10
Wagon R LXi Minor
VX (O) BL i-VTEC
Alto LXi
Magna 1.2
Crysta 2.4 VX 7 STR
2.0L TDI Comfortline
Alto LX BS-IV
GT TSI Sport
Ertiga VDi
4X4
Ritz Vxl (ABS) BS-IV
ZX i-VTEC
Camper 2WD
VX i-VTEC Opt
Dzire ZXi
SLX
Alto LXi BS-III
1.6 CRDI SX (O) AT
Swift Dzire VDi BS-IV
i20 Sportz 1.4 CRDi
S8 7 Seater
Ertiga ZDi
2.4 VTi L MT
FE 2 WD AT
Baleno Delta 1.2
1.5 VX i-DTEC
1.4L TSI Comfortline
Crysta 2.8 GX AT 7 STR
SV
Ertiga VXi
XLT 4X2
1.5 SV i-VTEC
W8
LTZ
Elegance 1.8 TSI AT
2.5 Z 7 SEATER
Ertiga VDI BS-IV
i20 Era 1.2
85 PS RXL
Titanium Diesel
S i-VTEC
110 PS RXZ
Swift DZire VDi
W8 4WD
1.0 RXT Opt
SX4 ZDi
RXL PETROL
Ritz LXi
Baleno Zeta AT 1.2
Crysta 2.4 G 8 STR
Sportz 1.2 Kappa2
Variant
0.0
0.5
1.0
1.5
2.0
2.5
3.0
count

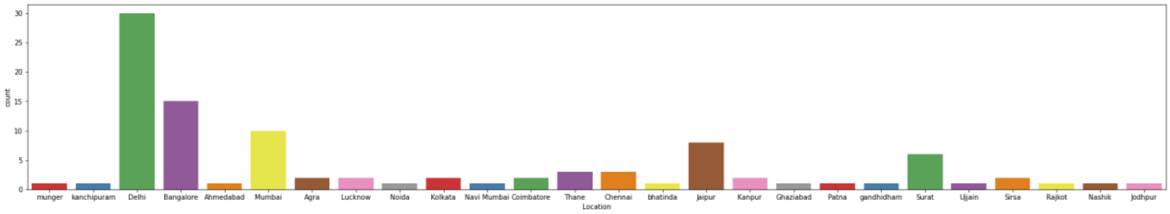




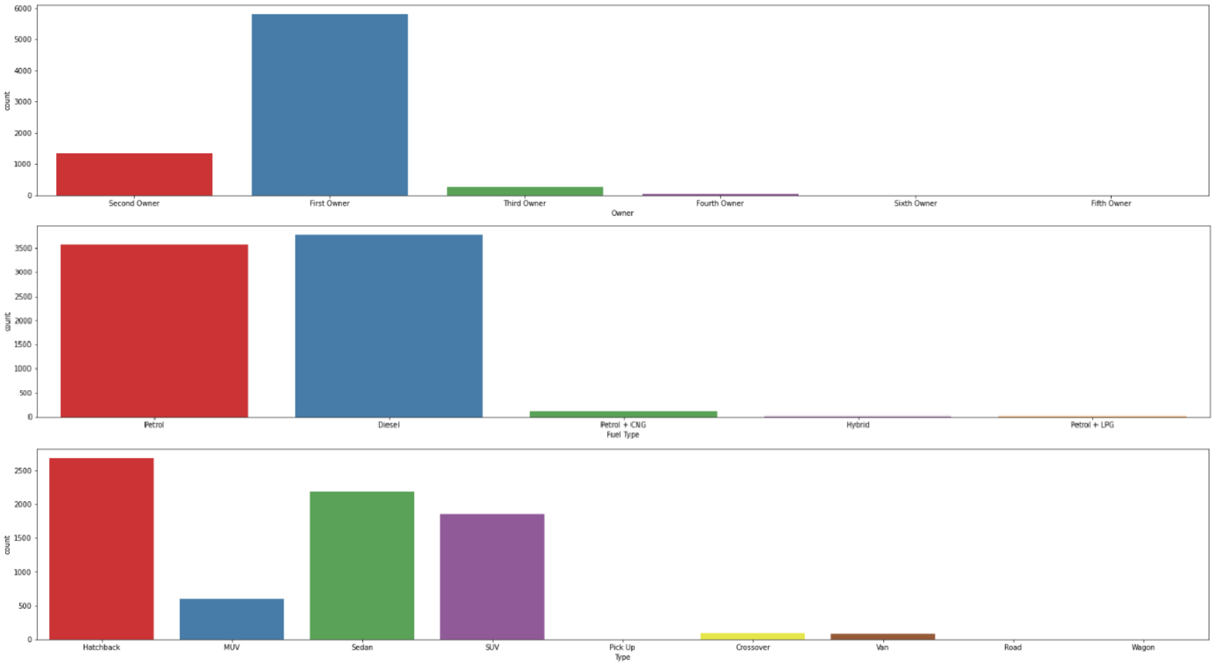
count
Maruti
Mahindra
Honda
Skoda
Nissan
Toyota
Volkswagen
Renault
Hyundai
Fiat
Tata
Ford
Chevrolet
Kia
Datsun
MG
Isuzu
Mitsubishi
Eicher
Mercedes-benz
Opel
Jeep
Brand



count
Manual
Automatic
Automated Manual Transmission
Transmission



count
munger
kanchipuram
Delhi
Bangalore
Ahmedabad
Mumbai
Agra
Lucknow
Noida
Kolkata
Navi Mumbai
Coimbatore
Thane
Chennai
bhatinda
Jaipur
Kanpur
Ghaziabad
Patna
gandhidham
Surat
Ujjain
Sirsa
Rajkot
Nashik
Jodhpur
Location

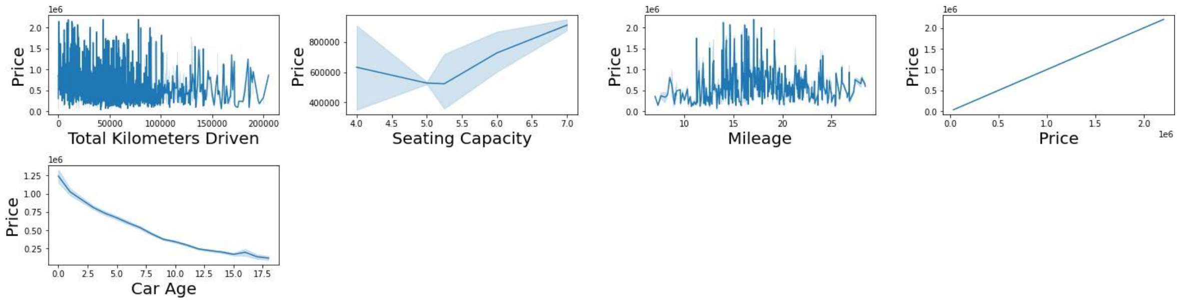


From the above graphs it is observed that:

- Swift VDi,Swift DZire VDi,Swift VXi,Wagon R LXi,Alto LXi are the most common used cars on sale
- Maruti and Hyundai are the most common brands of used cars
- Manual Transmission is the most common amongst cars
- Most used cars on sale are located in Delhi and Bangalore
- Most used cars on sale have only had 1 owner before
- Petrol and Diesel are the most common fuel types
- Hatchback,Sedan and SUV are the most common Car types available

Bivariate Analysis

Interpreting Relationship between Dependent Variable and Independent Variable Columns



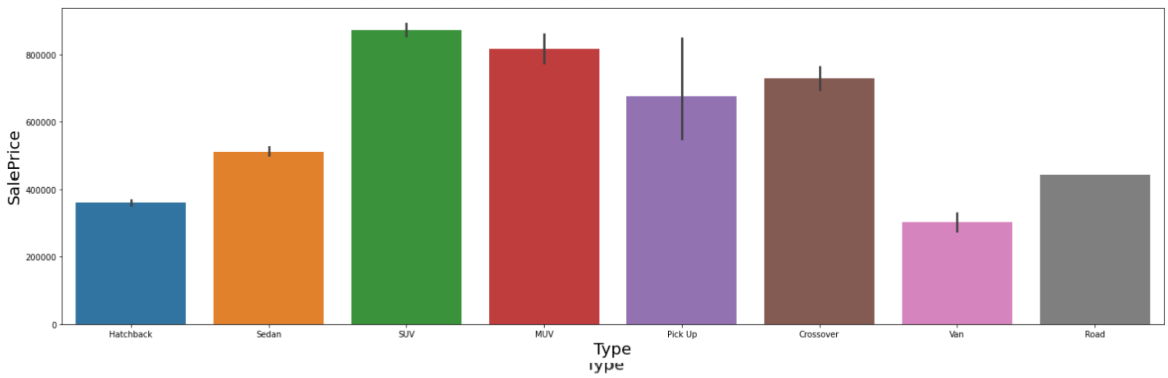
Following Observations are made from graphs above:

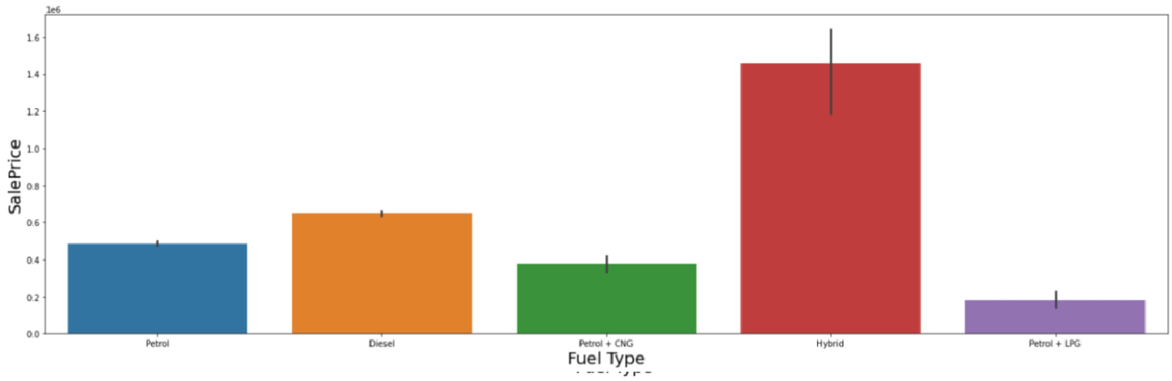
- There is a negative correlation between Total Kilometers Driven and Price

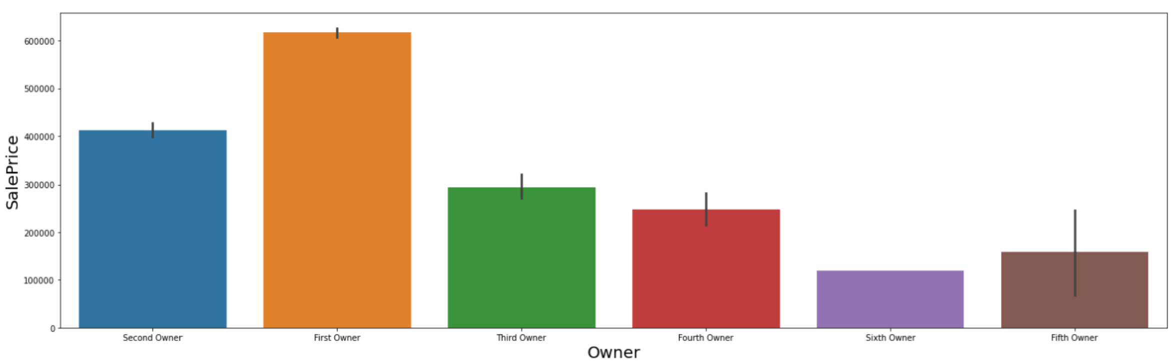
- Cars with Mileage between 14 km/l and 19 km/l have the highest prices

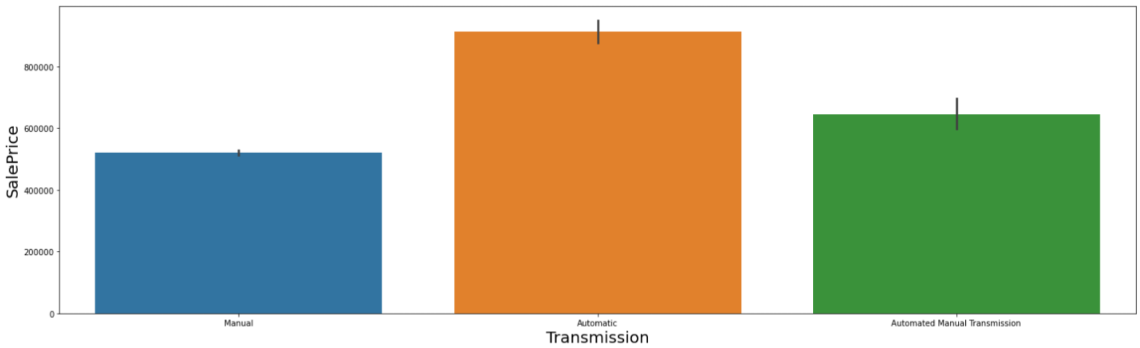
- There is a negative correlation between Car Age and Price

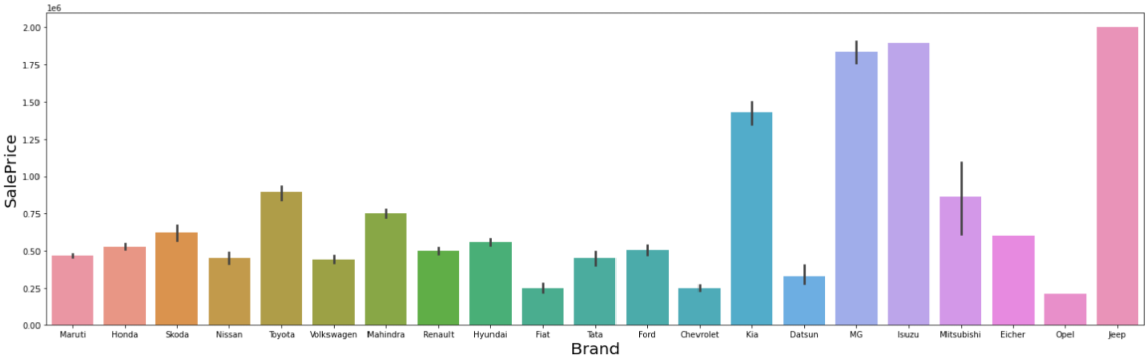
- There is a positive correlation between Seating capacity and Price

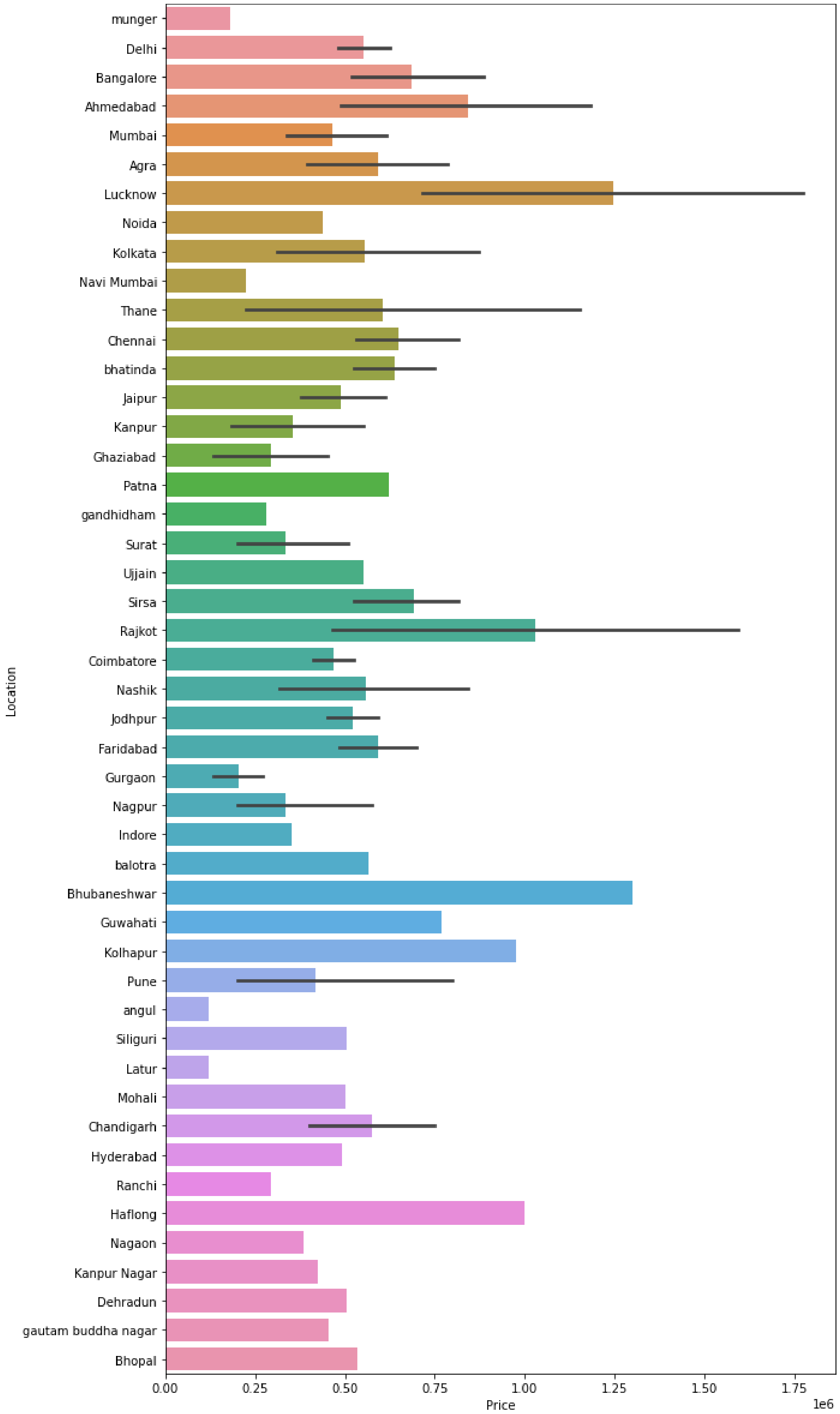


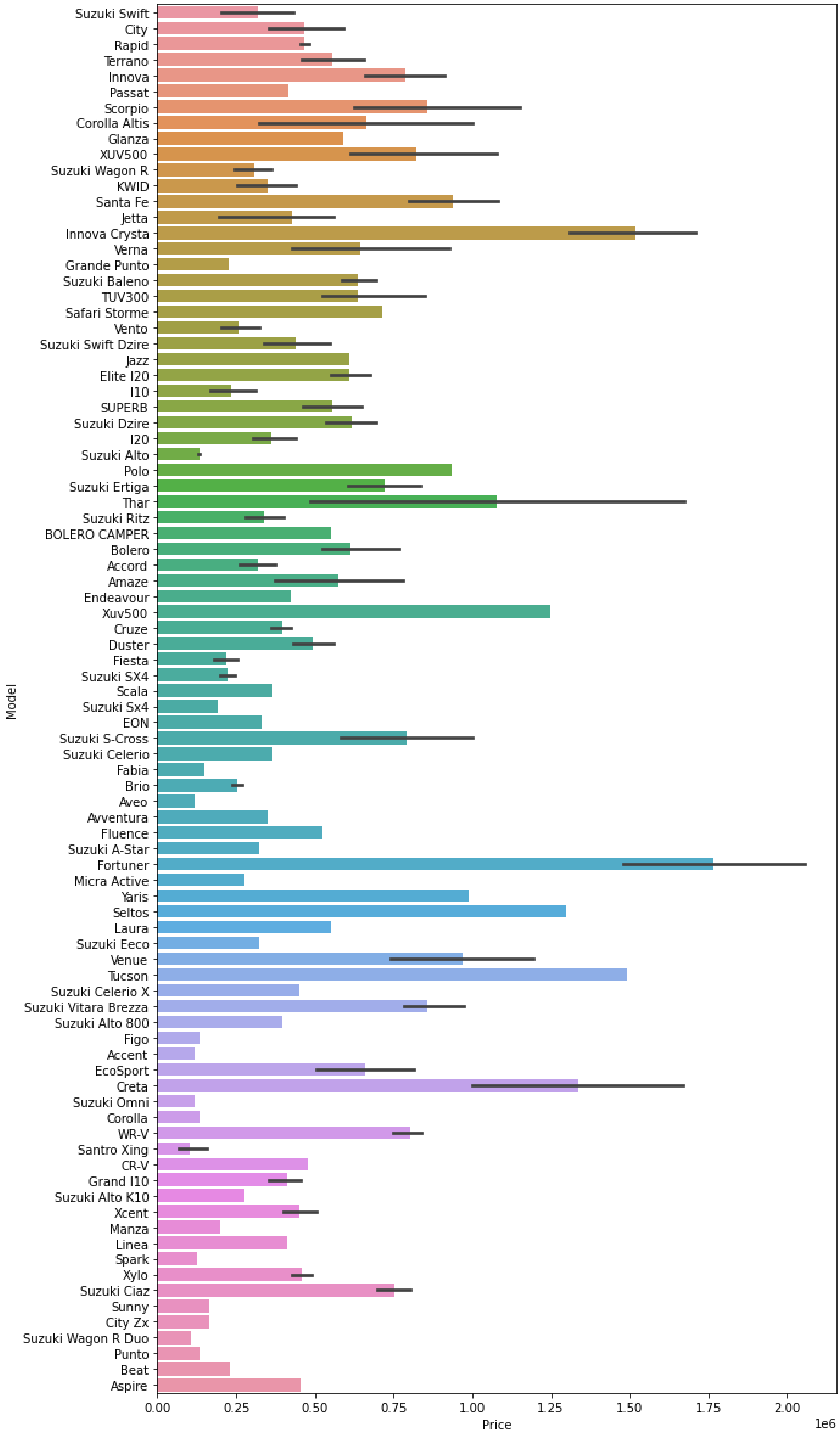





SalePrice
800000
600000
400000
200000
0
Manual
Automatic
Automated Manual Transmission
Transmission


1e6
SalePrice
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Maruti
Honda
Skoda
Nissan
Toyota
Volkswagen
Mahindra
Renault
Hyundai
Fiat
Tata
Ford
Chevrolet
Kia
Datsun
MG
Isuzu
Mitsubishi
Eicher
Opel
Jeep
Brand


munger
Delhi
Bangalore
Ahmedabad
Mumbai
Agra
Lucknow
Noida
Kolkata
Navi Mumbai
Thane
Chennai
bhatinda
Jaipur
Kanpur
Ghaziabad
Patna
gandhidham
Surat
Ujjain
Sirsa
Rajkot
Coimbatore
Nashik
Jodhpur
Faridabad
Gurgaon
Nagpur
Indore
balotra
Bhubaneshwar
Guwahati
Kolhapur
Pune
angul
Siliguri
Latur
Mohali
Chandigarh
Hyderabad
Ranchi
Haflong
Nagaon
Kanpur Nagar
Dehradun
gautam buddha nagar
Bhopal
Location
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
Price
1e6



Suzuki Swift
City
Rapid
Terrano
Innova
Passat
Scorpio
Corolla Altis
Glanza
XUV500
Suzuki Wagon R
KWID
Santa Fe
Jetta
Innova Crysta
Verna
Grande Punto
Suzuki Baleno
TUV300
Safari Storme
Vento
Suzuki Swift Dzire
Jazz
Elite I20
I10
SUPERB
Suzuki Dzire
I20
Suzuki Alto
Polo
Suzuki Ertiga
Thar
Suzuki Ritz
BOLERO CAMPER
Bolero
Accord
Amaze
Endeavour
Xuv500
Cruze
Duster
Fiesta
Suzuki SX4
Scala
Suzuki Sx4
EON
Suzuki S-Cross
Suzuki Celerio
Fabia
Brio
Aveo
Avventura
Fluence
Suzuki A-Star
Fortuner
Micra Active
Yaris
Seltos
Laura
Suzuki Eeco
Venue
Tucson
Suzuki Celerio X
Suzuki Vitara Brezza
Suzuki Alto 800
Figo
Accent
EcoSport
Creta
Suzuki Omni
Corolla
WR-V
Santro Xing
CR-V
Grand I10
Suzuki Alto K10
Xcent
Manza
Linea
Spark
Xylo
Suzuki Ciaz
Sunny
City Zx
Suzuki Wagon R Duo
Punto
Beat
Aspire
Model
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Price
1e6

Following Observations are made from graphs above:

- SUV,MUV,Pickup and Crossover type Cars have the highest Prices

- Hybrid Fuel Type Cars are the costliest

- As the Number of previous owners increases, the price of used car decreases, so there is a negative correlation between ownership and Car Price

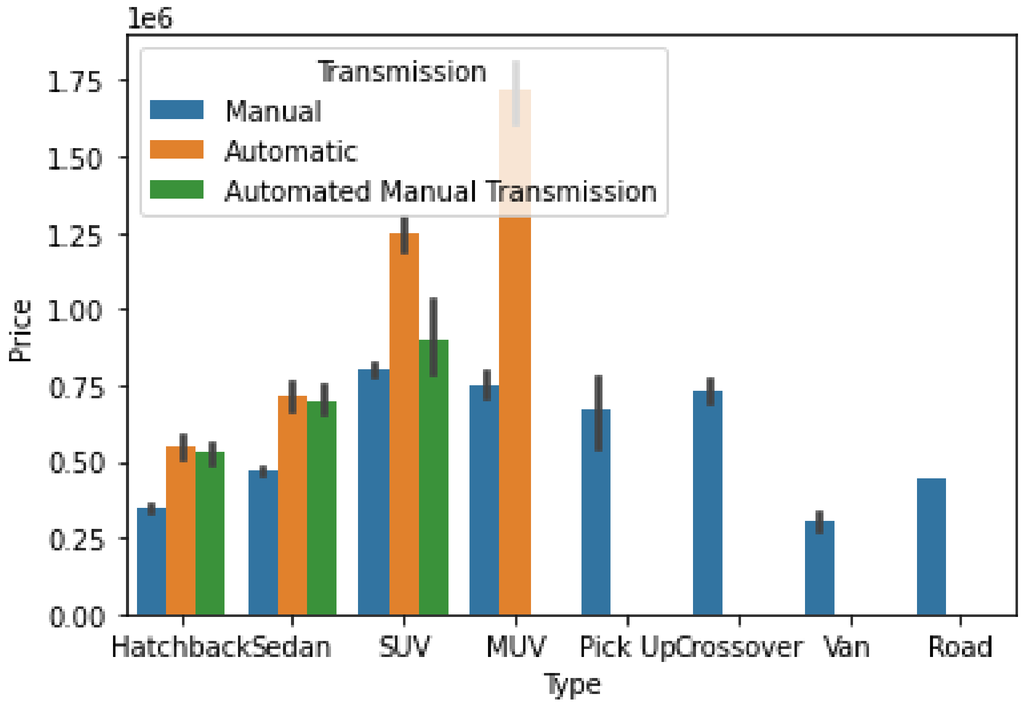
- Automatic Cars have the highest prices

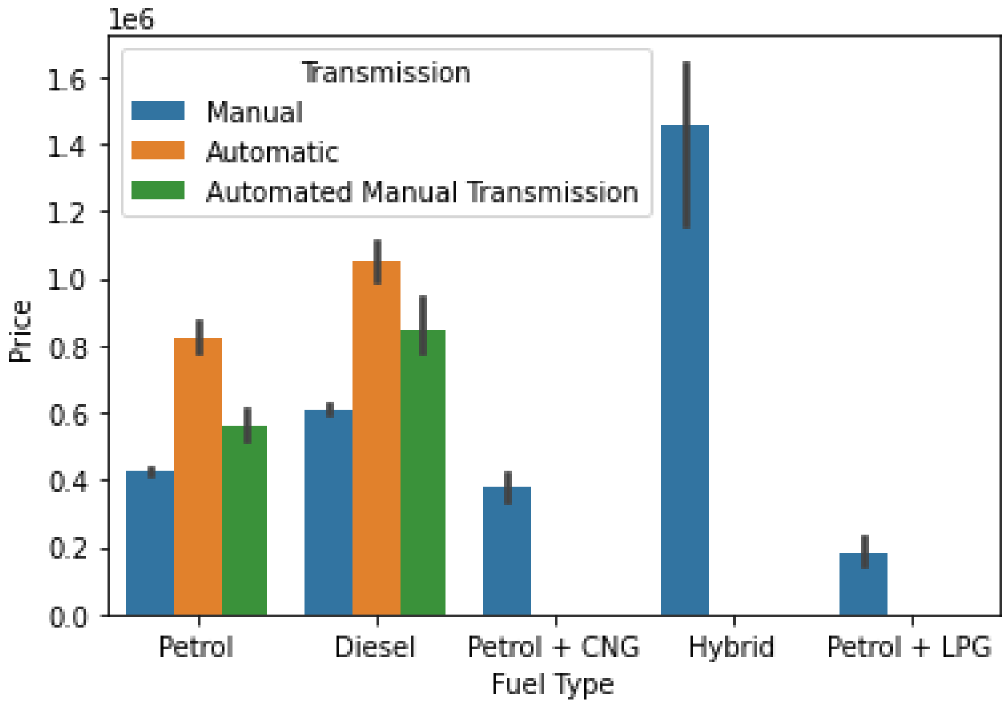
- Kia,MG,Isuzu and Jeep are amongst the most expensive Car Brands while Maruti,Volkswagen,Chevrolet,Opel,Tata,Honda,Fiat and Ford are the most affordable Car Brands

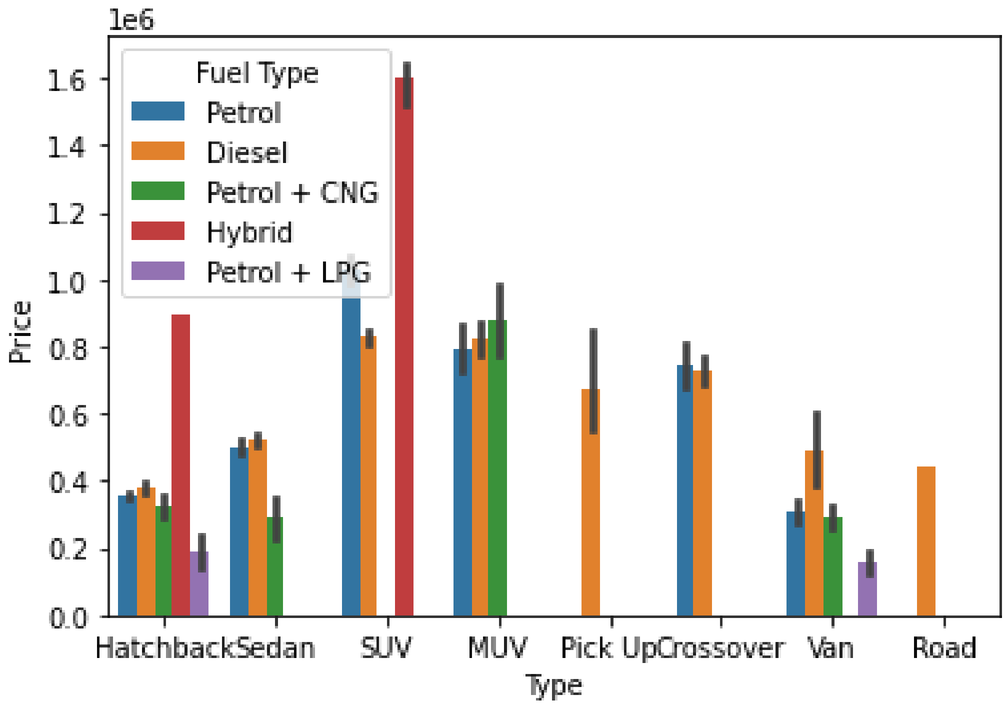
- Car Prices are highest in Lucknow,Rajkot,Bhubhaneshwar,Kohlapur,Ahmedabad and Haflong

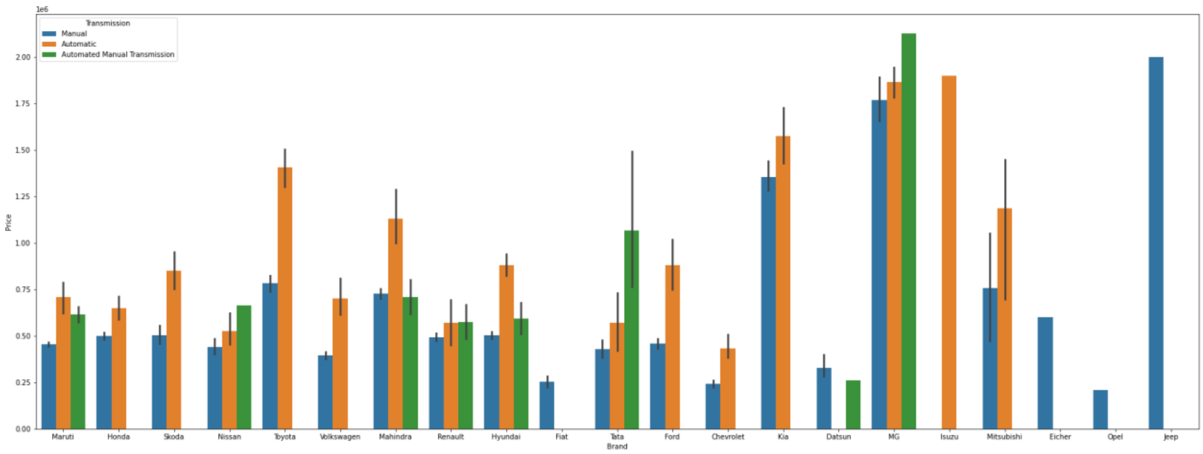
- Creta,Fortuner,Tucson,Seltos,Xuv500,Thar,Innova Crysta are amongst the most expensive car models, while Beat,Punto,Wagon R,Santro Xing,Alto,Fiesta are amongst the most affordable

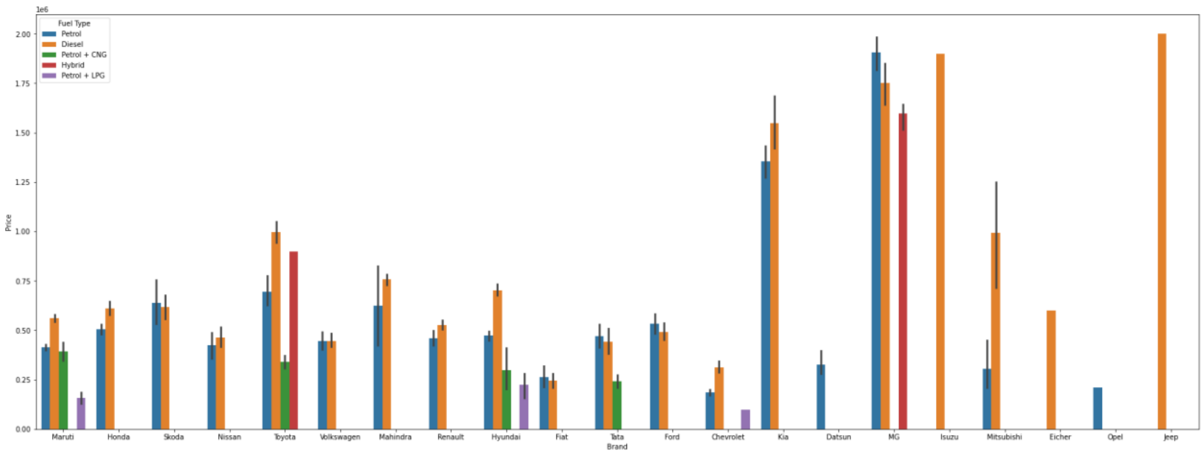
Multivariate Analysis






1e6
Fuel Type
Petrol
Diesel
Petrol + CNG
Hybrid
Petrol + LPG
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
Price
Hatchback
Sedan
SUV
MUV
Pick Up
Crossover
Van
Road
Type

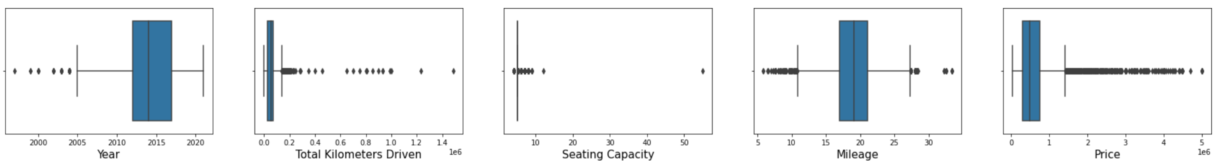

1e6
Transmission
Manual
Automatic
Automated Manual Transmission
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Price
Maruti
Honda
Skoda
Nissan
Toyota
Volkswagen
Mahindra
Renault
Hyundai
Fiat
Tata
Ford
Chevrolet
Kia
Datsun
MG
Isuzu
Mitsubishi
Eicher
Opel
Jeep
Brand


1e6
Fuel Type
Petrol
Diesel
Petrol + CNG
Hybrid
Petrol + LPG
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Price
Maruti
Honda
Skoda
Nissan
Toyota
Volkswagen
Mahindra
Renault
Hyundai
Fiat
Tata
Ford
Chevrolet
Kia
Datsun
MG
Isuzu
Mitsubishi
Eicher
Opel
Jeep
Brand

Following Observations are made from graphs above:

- SUV and MUV cars with Automatic Transmission are the costliest
- Hybrid Cars with Manual Transmission have the highest price followed by Diesel and Petrol cars with Automatic Transmission
- Hybrid Type Hatchback and SUV Cars are the most expensive
- Automatic Transmission Variants are the most expensive cars of most car brands
- Diesel Variants,followed by Petrol Variants are the most expensive cars of most car brands

Checking for Outliers



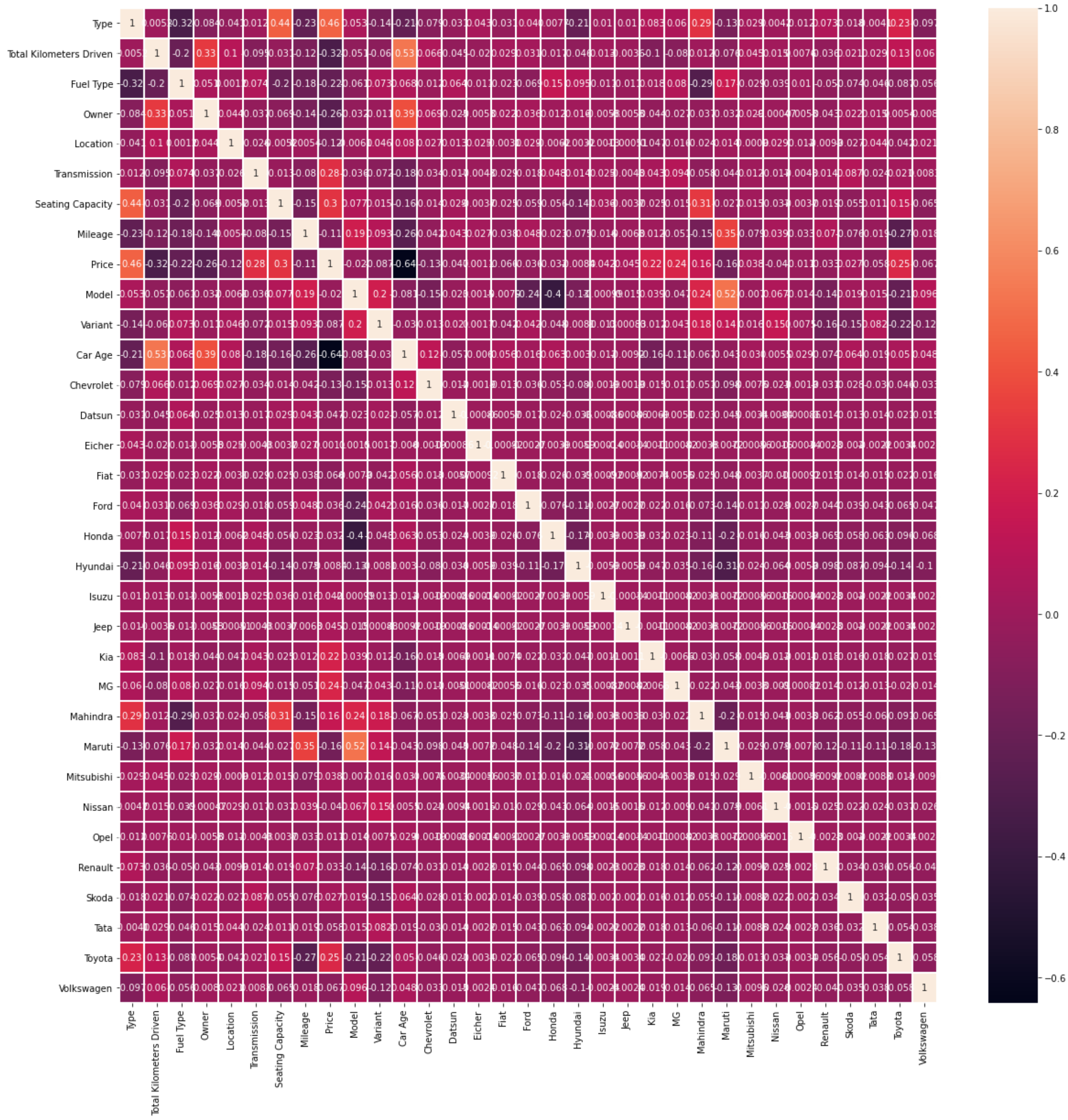
There are considerable outliers in the columns.

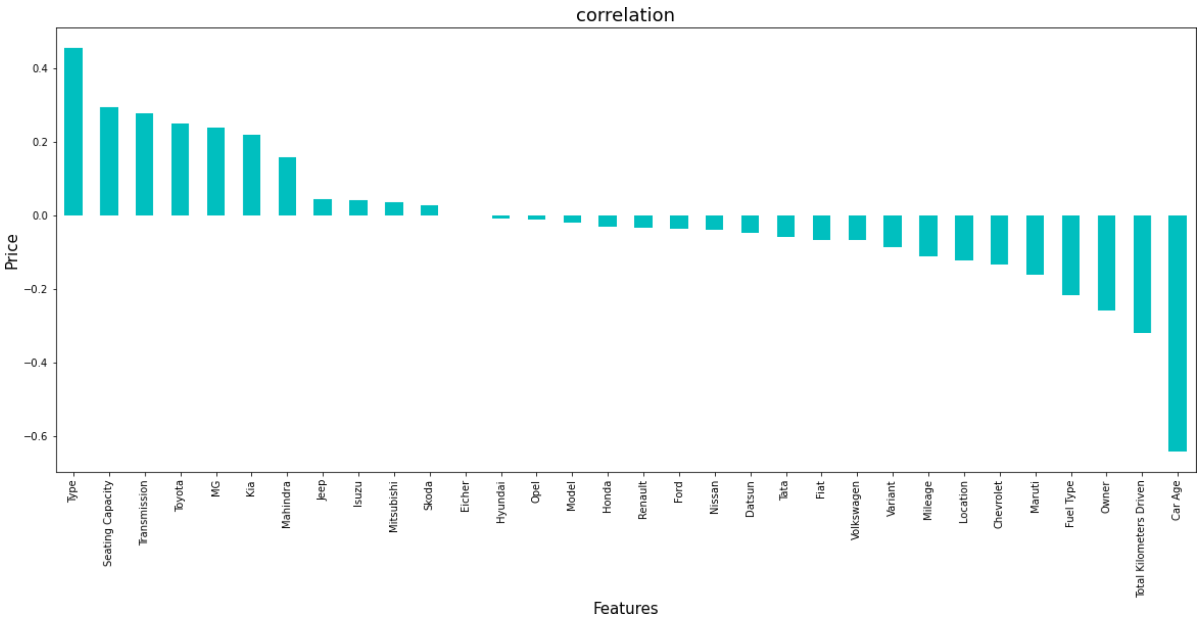
Outliers were Removed using Z score method which resulted in a total data loss of 3.82%, which is within acceptable range.

Encoding Categorical Columns

Categorical Columns were encoded using Label Encoding technique and get_dummies() technique.

Finding Correlation between Feature and Target columns





Type, Seating Capacity have the strongest positive correlation with Price while Car Age,Total Kilometers Driven,Owner and Fuel Type have the strongest negative correlation with Price.

Model/s Development and Evaluation

Feature Selection

Features were first checked for presence of multicollinearity and then based on respective ANOVA f-score values, the feature columns were
selected that would best predict the Target variable, to train and test machine learning models.

	Features	vif
0	Type	1.820311
1	Total Kilometers Driven	1.589312
2	Fuel Type	1.610300
3	Owner	1.224261
4	Location	1.018970
5	Transmission	1.091238
6	Seating Capacity	1.378541
7	Mileage	1.785701
8	Model	2.356090
9	Variant	1.291241
10	Car Age	1.929347
11	Chevrolet	inf
12	Datsun	inf
13	Eicher	inf

There is no Multicollinearity among the columns

```python
from sklearn.feature_selection import SelectKBest, f_classif
```

```python
bestfeat = SelectKBest(score_func = f_classif, k = 'all')
fit = bestfeat.fit(X,y)
dfscores = pd.DataFrame(fit.scores_)
dfcolumns = pd.DataFrame(X.columns)
```

```python
fit = bestfeat.fit(X,y)
dfscores = pd.DataFrame(fit.scores_)
dfcolumns = pd.DataFrame(X.columns)
dfcolumns.head()
featureScores = pd.concat([dfcolumns,dfscores],axis = 1)
featureScores.columns = ['Feature', 'Score']
print(featureScores.nlargest(75,'Score'))
```

```
                     Feature     Score
21                        MG  7.810969
20                       Kia  6.621769
10                   Car Age  6.409601
6           Seating Capacity  2.360881
0                       Type  2.233844
1    Total Kilometers Driven  2.005306
30                    Toyota  1.805455
22                  Mahindra  1.636365
5               Transmission  1.631383
2                  Fuel Type  1.622886
12                   Datsun   1.333130
4                   Location  1.330697
29                      Tata  1.282645
3                      Owner  1.260928
24                Mitsubishi  1.198383
7                    Mileage  1.172318
23                    Maruti  1.142386
9                    Variant  1.086801
17                   Hyundai  1.048762
28                     Skoda  1.043426
8                      Model  1.032035
16                     Honda  0.962018
31                Volkswagen  0.926324
11                 Chevrolet  0.908046
```

Using SelectKBest and f_classif for measuring the respective ANOVA f-score values of the columns, the best features were selected. Using StandardScaler, the features were scaled by resizing the distribution values so that mean of the observed values in each feature column is 0 and standard deviation is 1. From sklearn.model_selection's train_test_split, the data was divided into train and test data. Training data comprised 75% of total data where as test data comprised 25% based on the best random state that would result in best model accuracy.

The model algorithms used were as follows:

- Ridge: Ridge regression is a model tuning method that is used to analyse any data that suffers from multicollinearity. This method performs L2 regularization. Since the features have multicollinearity occurs, least-squares are unbiased, and variances are large, this results in predicted values to be far away from the actual values. Ridge shrinks the parameters. Therefore, it is used to prevent multicollinearity.

- DecisionTreeRegressor: Decision Tree solves the problem of machine learning by transforming the data into a tree representation. Each internal node of the tree representation denotes an attribute and each leaf node denotes a class label. A decision tree does not require normalization of data. A decision tree does not require normalization of data.
- XGBRegressor: XGBoost uses decision trees as base learners; combining many weak learners to make a strong learner. As a result it is referred to as an ensemble learning method since it uses the output of many models in the final prediction. It uses the power of parallel processing,supports regularization, and works well in small to medium dataset.
- RandomForestRegressor: A random forest is a meta estimator that fits a number of classifying decision trees on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. A random forest produces good predictions that can be understood easily. It reduces overfitting and can handle large datasets efficiently. The random forest algorithm provides a higher level of accuracy in predicting outcomes over the decision tree algorithm.
- Support Vector Regressor: SVR works on the principle of SVM with few minor differences. Given data points, it tries to find the curve. But since it is a regression algorithm instead of using the curve as a decision boundary it uses the curve to find the match between the vector and position of the curve. Support Vectors helps in determining the closest match between the data points and the function which is used to represent them. SVR is robust to the outliers. SVR performs lower computation compared to other regression techniques.
- K-Nearest Neighbor Regressor: It is a lazy learning, non-parametric algorithm. It uses data with several classes to predict the classification of the new sample point. KNN is non-parametric since it doesn’t make any assumptions on the data being studied. The Training phase is fast. KNN Regressor keeps all training data since they are needed

during testing phase. KNN algorithm fairs across all parameters of considerations. But mostly, it is used due to its ease of interpretation and low calculation time.

Regression Model Building

Finding the Best Random State

```
from sklearn.ensemble import RandomForestRegressor
maxAcc = 0
maxRS=0
for i in range(1,100):
    x_train,x_test,y_train,y_test = train_test_split(scaled_x_best,y,test_size = .25, random_state = i)
    modRF =  RandomForestRegressor()
    modRF.fit(x_train,y_train)
    pred = modRF.predict(x_test)
    acc  = r2_score(y_test,pred)
    if acc>maxAcc:
        maxAcc=acc
        maxRS=i
print(f"Best Accuracy is: {maxAcc} on random_state: {maxRS}")
```

```
Best Accuracy is: 0.9122189321478518 on random_state: 90
```

```
x_train,x_test,y_train,y_test = train_test_split(scaled_x_best,y,test_size = .25, random_state =90)
```

```
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import Ridge
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from xgboost import XGBRegressor
from sklearn.svm import SVR
from sklearn.neighbors import KNeighborsRegressor

```

```
from sklearn.metrics import r2_score,mean_squared_error
```

```
rf = RandomForestRegressor()
dt = DecisionTreeRegressor()
xg = XGBRegressor()
SV= SVR()
r=Ridge()
KNN = KNeighborsRegressor()
```

Training the Models

```
rf.fit(x_train,y_train)
xg.fit(x_train,y_train)
SV.fit(x_train,y_train)
r.fit(x_train,y_train)
dt.fit(x_train,y_train)
KNN.fit(x_train,y_train)
```

```
KNeighborsRegressor()
```

All models have been trained.

Ridge Regression Model

```
y_r_pred = r.predict(x_test)
```

R2 Score

```
r2_score(y_test,y_r_pred)
```

```
0.7037453679274228
```

Mean Squared Error

```
mean_squared_error(y_test,y_r_pred)
```

```
43531846036.22704
```

Root Mean Squared Error

```
np.sqrt(mean_squared_error(y_test,y_r_pred))
```

```
208642.86720668655
```

Random Forest Regression Model

```
y_rf_pred = rf.predict(x_test)
```

R2 Score

```
r2_score(y_test,y_rf_pred)
```

```
0.9060733783304681
```

Mean Squared Error

```
mean_squared_error(y_test,y_rf_pred)
```

13801638153.692486

Root Mean Squared Error

```
np.sqrt(mean_squared_error(y_test,y_rf_pred))
```

117480.37348294602

XGB Regression Model

```
y_xg_pred = xg.predict(x_test)
```

R2 Score

```
r2_score(y_test,y_xg_pred)
```

0.9263521747648791

Mean Squared Error

```
mean_squared_error(y_test,y_xg_pred)
```

10821858772.668312

Root Mean Squared Error

```
np.sqrt(mean_squared_error(y_test,y_xg_pred))
```

104028.16336294856

Support Vector Regression Model

```
y_svr_pred = SV.predict(x_test)
```

R2 Score

```
r2_score(y_test,y_svr_pred)
```

-0.06863993854979955

Mean Squared Error

```
mean_squared_error(y_test,y_svr_pred)
```

157026639373.24176

Root Mean Squared Error

```
np.sqrt(mean_squared_error(y_test,y_svr_pred))
```

396265.8695538158

Decision Tree Regression Model

```
y_dt_pred = dt.predict(x_test)
```

R2 Score

```
r2_score(y_test,y_dt_pred)
```

0.8361681075174812

Mean Squared Error

```
mean_squared_error(y_test,y_dt_pred)
```

24073563574.27525

Root Mean Squared Error

```
np.sqrt(mean_squared_error(y_test,y_dt_pred))
```

155156.5776055764

KNN Regression Model

```
y_knn_pred = KNN.predict(x_test)
```

R2 Score

```
r2_score(y_test,y_knn_pred)
```

0.8055276499903519

R2 Score

```
r2_score(y_test,y_knn_pred)
```

0.8055276499903519

Mean Squared Error

```
mean_squared_error(y_test,y_knn_pred)
```

28575892095.585197

Root Mean Squared Error

```
np.sqrt(mean_squared_error(y_test,y_knn_pred))
```

169044.05371259054

Analyzing Accuracy of The Models

Mean Squared Error and Root Mean Squared Error metrics were used to evaluate the Model performance. The advantage of MSE and RMSE being that it is easier to compute the gradient. As, we take square of the error, the effect of larger errors become more pronounced than smaller error, hence the model can now focus more on the larger errors.

Cross validation is a technique for assessing how the statistical analysis generalises to an independent data set.It is a technique for evaluating machine learning models by training several models on subsets of the available input data and evaluating them on the complementary subset of the data.

Using cross-validation, there are high chances that we can detect over-fitting with ease. Model Cross Validation scores were then obtained for assessing how the statistical analysis generalises to an independent data set. The models were evaluated by training several models on subsets of the available input data and evaluating them on the complementary subset of the data.

```
from sklearn.model_selection import ShuffleSplit,cross_val_score
```

Ridge Regression

```
cross_val_score(r,scaled_x_best,y,cv=5).mean()
```

```
0.6884482081366692
```

Random Forest Regression

```
cross_val_score(rf,scaled_x_best,y,cv=5).mean()
```

```
0.8912375239032073
```

XGB Regression

```
cross_val_score(xg,scaled_x_best,y,cv=5).mean()
```

```
0.9092138998450912
```

SV Regression

```
cross_val_score(SV,scaled_x_best,y,cv=5).mean()
```

```
-0.05723565341101864
```

Decision Tree Regression

```
cross_val_score(dt,scaled_x_best,y,cv=5).mean()
```

```
0.8148761573341898
```

KNN Regression

```
cross_val_score(KNN,scaled_x_best,y,cv=5).mean()
```

```
0.8024696110352441
```

Interpretation of the Results

Based on comparing Accuracy Score results with Cross Validation results, it is determined that XGB Regressor is the best model. It also has the lowest Root Mean Squared Error score.

Hyper Parameter Tuning

GridSearchCV was used for Hyper Parameter Tuning of the XGB Regressor model.

Hyper Parameter Tuning

```
from sklearn.model_selection import GridSearchCV
```

XGB Regressor

```
parameter = {'booster':["gbtree","gblinear"],'eta': [0.01,0.1,0.2,0.3,0.5,1],'min_child_weight':np.arange(6),'max_depth':[10
```

```
GridCV = GridSearchCV(XGBRegressor(),parameter,cv=5,n_jobs = -1,verbose = 1)
```

```
GridCV.fit(x_train,y_train)
```

```
Fitting 5 folds for each of 1296 candidates, totalling 6480 fits
```

```
GridCV.best_params_
```

```
{'booster': 'gbtree',
 'eta': 0.1,
 'max_depth': 10,
 'min_child_weight': 2,
 'subsample': 0.5}
```

```
Best_mod = XGBRegressor(booster = 'gbtree',eta = 0.1, max_depth= 10, min_child_weight = 2,subsample = 0.5)
Best_mod.fit(x_train,y_train)
```

```
XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,
             colsample_bynode=1, colsample_bytree=1, eta=0.1, gamma=0,
             gpu_id=-1, importance_type='gain', interaction_constraints='',
             learning_rate=0.100000001, max_delta_step=0, max_depth=10,
             min_child_weight=2, missing=nan, monotone_constraints='()',
             n_estimators=100, n_jobs=16, num_parallel_tree=1, random_state=0,
             reg_alpha=0, reg_lambda=1, scale_pos_weight=1, subsample=0.5,
             tree_method='exact', validate_parameters=1, verbosity=None)
```

```
xgbpred = Best_mod.predict(x_test)

acc = r2_score(y_test,xgbpred)
print(acc*100)
```

```
92.4828003021342
```

Based on the input parameter values and after fitting the train datasets The XGB Regressor model was further tuned based on the parameter values yielded from GridsearchCV. The Random Forest Regressor model displayed an accuracy of 92.48%

This model was then tested using a scaled Test Dataset comprising of 7207 entries. The model performed with good amount of accuracy.

```
Prediction_accuracy = pd.DataFrame({'Predictions': mod.predict(scaled_x_best), 'Actual Values': y[0:7207]})
Prediction_accuracy
```

	Predictions	Actual Values
0	1.798894e+05	180000.0
1	3.631930e+05	350000.0
2	4.885275e+05	480000.0
3	6.933516e+05	654050.0
4	9.021162e+05	850000.0
...	...	...
7202	6.616298e+05	850000.0
7203	7.306963e+05	715000.0
7204	1.333314e+06	1299997.0
7205	2.090563e+05	210000.0
7206	1.699480e+06	1645000.0

In summary, Based on the visualizations of the feature-column relationships, it is determined that, Features like Type, Seating Capacity have the strongest positive correlation with Price while Car Age,Total Kilometers Driven,Owner and Fuel Type have the strongest negative correlation with Price. and are some of the most important features to predict the label values. XGB Regressor Performed the best out of all the models that were tested. It also worked well with the outlier handling.

CONCLUSION

Key Findings and Conclusions of the Study and Learning Outcomes with respect to Data Science

Based on the in-depth analysis of the Housing Project, The Exploratory analysis of the datasets, and the analysis of the Outputs of the models the following observations are made:

• Car attributes like Type,Car Age, Seating Capacity,Total Kilometers Driven,Transmission,Fuel Type,Owner and Mileage etc play a big role in influencing the used car price.

• Brand Name also has a very important role in determining the used car price.

• Various plots like Barplots,Countplots and Lineplots helped in visualising the Feature-label relationships which corroborated the importance of Car features and attributes for estimating Sale Prices.

• Due to the Training dataset being very small, the outliers had to be retained for proper training of the models.

• Therefore, XGB Regressor, which uses the power of parallel processing,supports regularization, and works well in small to medium dataset performed well despite having to work on small dataset.

Learning Outcomes of the Study in respect of Data Science

Data cleaning was a very important step in removing plenty of anomalous data from the huge dataset that was provided. Visualising data helped identify outliers and the relationships between target and feature columns as well as analysing the strength of correlation that exists between them.

Limitations of this work and Scope for Future Work

A small dataset to posed a challenge in building highly accurate models. The presence of anomalous entries in the numbers heavily distorted the data distributions and may have had some impact on model learning.

Availability of more features would help build better models.