



Used Car: Price-Prediction



Submitted by:
Deepak kr. Singh

ACKNOWLEDGMENT

This includes mentioning of all the references, research papers, data sources, professionals and other resources that helped you and guided you in completion of the project.

- I would like to thank FlipRobo Technologies for providing me this opportunity and guidance throughout the project and all the steps that are implemented.
- I have primarily referred to various articles scattered across various websites for the purpose of getting an idea on Car price prediction (used car price prediction) project.
- I would like to thank the technical support team also for helping me out and reaching out to me on clearing all my doubts as early as possible.
- I would like to thank my project SME M/S Sapna Verma for providing the flexibility in time and also for giving us guidance in creating the project.
- I have referred to various articles in Towards Data Science and Kaggle



INTRODUCTION

Web scraping using website Cars24 and data of scraped price of a used car is a challenging task, due to the many factors that drive a used vehicle's price on the market. The focus of this project is developing machine learning models that can accurately predict the price of a used car based on its features, in order to make informed purchases. We implement and evaluate various learning methods on a dataset consisting of the sale prices of different makes and models across cities in the India. Our results show that Random Forest model and K-Means clustering with linear regression yield the best results, but are compute heavy. Conventional linear regression also yielded satisfactory results, with the advantage of a significantly lower training time in comparison to the aforementioned methods.

Because of new computing technologies, machine learning today is not like machine learning of the past. It was born from pattern recognition and the theory that computers can learn without being programmed to perform specific tasks; researchers interested in artificial intelligence wanted to see if computers could learn from data. The iterative aspect of machine learning is important because as models are exposed to new data, they are able to independently adapt. They learn from previous computations to produce reliable, repeatable decisions and results.

- **Business Problem Framing**

Car has become basic requirement of every person in this fast and dynamic world. It's a science that's not new – but one that has gained fresh momentum. While there is an end number of applications of machine learning in real life one of the most prominent application is the prediction problems. There are various topics on which the prediction can be applied. One such application is what this project is focused upon. Websites recommending items you might like based on previous purchases are using machine learning to analyse your buying history – and promote other items you'd be interested in. This ability to capture data, analyze it and use it to personalize a shopping experience (or implement a marketing campaign) is the future of retail.

The requirement of model building is to predict the price of car in accordance with variable and how the variable describe the car and help people to purchase used cars.

Business goal:

It's require to build a model for used cars with the independent variable given on website. This model will then used by the management to understand how exactly the price vary with the variable and what's the requirement of customers. According to that they can manipulate the strategy of the firm and concentrate on area that will yield high return. Further than the model will be good for management to understand the pricing or the requirement of this dynamic new market.

• Conceptual Background of the Domain Problem

- Used cars selling more than new cars in India! To be world's third largest market. **More consumers are looking at second-hand entry-level cars as opposed to buying new cars, this has taken a major chunk out of new car sales. A trend that is expected to grow even more!**

Estimating the sale prices of used cars is one of the basic project in Data Science. By finishing this article, i will be able to predict continuous variables using various types of linear regression algorithm: (Linear regression is an algorithm used to predict values that are continuous in nature. It became more popular because it is the best algorithm to start with if you are a newbie to ML.)

Technical Requirements:

- Scraped data contains over 5k entries each having various variables.
- Data contains Null values.Data treatment using domain knowledge, understanding.
- Extensive EDA has to be performed to gain relationships of important variable and price.
- Data contains numerical as well as categorical variable, handle them accordingly.
- Machine Learning models, apply regularization and determine values of Hyper Parameters.
- You need to find important features which affect the price positively or negatively.
- Two datasets are being provided (test.csv, train.csv), train model on train.csv dataset and predict on test.csv file.

• Review of Literature

The various applications and methods which inspired us to build our project. We did a background survey regarding the basic ideas of our project and used those ideas for the collection of information like the technological stack, algorithms, and shortcomings of our project which led us to build a better project.

CARS24 is web platform used for by me to scrape data for used car. Cars24 is a web platform where seller can sell their used car. It is an Indian Start-up with a simplified user interface which asks seller parameters like car model, kilometres travelled, year of registration and vehicle type (petrol, diesel, Petrol+LPG, Petrol+CNG). These allow the web model to run certain algorithms on given parameters and predict the price.

Vehicle Price : Get Vehicle Price just on android app or website which works on similar parameters as of Cars24. This app predicts vehicle prices on various parameter like Location, Model, Price and kilometres travelled. This app uses a machine learning approach to predict the price of a car, on the basis of fuel type petrol, diesel, electric vehicle or hybrid vehicle. App can predict the price of any vehicle because of the smartly optimized algorithm.

• Motivation for the Problem Undertaken

Deciding whether a used car is worth the posted price when you see listings online can be difficult. Several factors, including mileage, make, model, year, etc. can influence the actual worth of a car. From the perspective of a seller, it is also a dilemma to price a used car appropriately [2-3]. Based on existing data, the aim is to use machine learning algorithms to develop models for predicting used car prices.

Analytical Problem Framing

- Mathematical/ Analytical Modeling of the Problem

- I am working with the used car price dataset that contains various features and information about the cars and its sale price. Using the scraped data in form of 'read_csv' function provided by the Pandas package, which can import the data into our python environment. After importing the data, I have used the 'head' function to get a glimpse of our dataset. Data has been scraped from website Cars24.
- As used car are growing in market and there is huge demand for it, because car has become one of the fundamental essential of every human
- Hence the reason for continuous research in this sector. This project simply examines a dataset, which consists 5000+ observations and 10 features that contribute to the sale price of the cars. Dataset was cleaned and transformed and some explorations were done on it to answer some basic questions that anybody would like to ask about cars.
- Our dataset is ready in the right form with the right variables to be used in the algorithms, which results in improved model accuracy. Different ensemble algorithms were used on the dataset in this project. The overall result of this project shows that the most important variables that determine the price of a car being sold.

Keywords: cars price, Analysis, encoding, Ensemble Algorithms and Feature Engineering.

- Data Sources and their formats

- The dataset has been scraped from website Cars24 this data is only for academic use, not for any commercial.
- The dataset describe data related to cars with 5000+ records.
- The dataset is in csv. Format which contains train and test data.
- This dataset is to use simply examines data, which consists 5000+ observations and 10 features for model predication.
- The dataset is in both numerical as well as categorical data.

```
# For Bangalore
driver_B = webdriver.Chrome(r"C:\driver\chromedriver.exe")
driver_B.get('https://www.cars24.com/buy-used-car?sort=P&storeCityId=4709&pinId=560001')
ScrollNumber_B = 80
for i in range(1,ScrollNumber_B):
    driver_B.execute_script("window.scrollTo(1,50000000)")
    time.sleep(5)
file_B = open('cars24_Bangalore.html', 'w', encoding='utf-8')
file_B.write(driver_B.page_source)
file_B.close()
driver_B.close()

data_B = open('cars24_Bangalore.html','r')
soup_B = BeautifulSoup(data_B, 'html.parser')
```

- Data Preprocessing Done

In order to get a better understanding of the data, we plotted a histogram of the data. We noticed that the dataset had many outliers, primarily due to large price sensitivity

of used cars. Typically, models that are the latest year and have low mileage sell for a premium, however, there were many data points that did not conform to this. This is because accident history and condition can have a significant effect on the car's price. Since we did not have access to vehicle history and condition, we pruned our dataset to three standard deviations around the mean in order to remove outliers. We converted the Make, Model and State into one-hot vectors. Since we had over 5000+ cars in the dataset, we replaced the string representing the city with a boolean which was set if the population of the city was above a certain threshold i.e. a major city were the steps followed for the cleaning of the data? What were the assumptions done and what were the next actions steps over that?

• Data Inputs- Logic- Output Relationships

There are many website for used cars but this data has been scraped from website cars24

```
# For Bangalore
driver_B = webdriver.Chrome(r"C:\driver\chromedriver.exe")
driver_B.get('https://www.cars24.com/buy-used-car?sort=P&storeCityId=4709&pinId=560001')
ScrollNumber_B = 80
for i in range(1, ScrollNumber_B):
    driver_B.execute_script("window.scrollTo(1,50000000)")
    time.sleep(5)
file_B = open('cars24_Bangalore.html', 'w', encoding='utf-8')
file_B.write(driver_B.page_source)
file_B.close()
driver_B.close()
```

```
data_B = open('cars24_Bangalore.html', 'r')
soup_B = BeautifulSoup(data_B, 'html.parser')
```

Storing the DataFrame to a 'CSV' file

```
df.to_csv("Cars24_car.csv")
```

Storing the dataset for further analysis

Loading the DataFrame

```
df = pd.read_csv("Cars24_car.csv")
df.drop('Unnamed: 0', axis=1, inplace=True)
df.head()
```







	Car Brand	Model	Price	Model Year	Location	Fuel	Driven (Kms)	Gear	Ownership	EMI (monthly)
0	Hyundai	Elite i20SPORTZ 1.4	687499	2016	Hyderabad	Diesel	51885	Manual	1	15293
1	Maruti	ErigaZXI	648099	2014	Hyderabad	Petrol	67232	Manual	1	14430
2	Tata	TiagoXZ 1.2 REVOTRON	481999	2017	Hyderabad	Petrol	28702	Manual	2	10722
3	Maruti	New Wagon-R 1.0 VXI (O)	538999	2019	Hyderabad	Petrol	19377	Manual	1	11945
4	Skoda	OctaviaStyle 1.4 TSI MT	1292899	2017	Hyderabad	Petrol	47264	Manual	1	28780

Exploratory Data Analysis (EDA)

- This section shows the exploration done on the dataset, which is what motivated the use of the algorithm. The following are the questions explored in this project and for the sake of writing I will only show some of the visuals here while I will provide the codes that shows the full visualization of all the questions explored.
- Is there a significant relationship between sale price and car model? It was used to check for this and we can see that there is a relationship between how much old cars are and how much is its selling price.
- There are two primary phases in the system:
 1. Training phase: The system is trained by using the data in the data set and fits a model (line/curve) based on the algorithm chosen accordingly.

2. Testing phase: the system is provided with the inputs and is tested for its working. The accuracy is checked. And therefore, the data that is used to train the model or test it, has to be appropriate. The system is designed to detect and predict price of used car and hence appropriate algorithms must be used to do the two different tasks. Before the algorithms are selected for further use, different algorithms were compared for its accuracy. The well-suited one for the task was chosen.

Steps :

-  Importing the required packages into our python environment
-  Importing the car price data and do some EDA on it
-  Data Visualization on the car price data
-  Feature Selection & Data Split
-  Modelling the data using the algorithms
-  Evaluating the built model using the evaluation metrics

- State the set of assumptions (if any) related to the problem under consideration
- Finally, we conclude which model is best suitable for the given case by evaluating each of them using the evaluation metrics provided by the scikit-learn package. This model will help to decide the car price prediction.

This model shows the Used cars selling more than new cars in India & largest market.

● Hardware and Software Requirements and Tools Used

Listing down the hardware and software requirements along with the tools, libraries and packages used.

Windows 10 64bit

Anaconda 2021 / Python version – Python 3.9.5

Software: Jupyter notebook, Python, Panda library, numpy library, Matplotlib library, Seaborn library



Python: Python is a general-purpose, and high-level programming language which is best known for its efficiency and powerful functions. Its ease to use, which makes it more accessible. Python provides data scientists with an extensive amount of tools and packages to build machine learning models. One of its special features is that we can build various machine learning with less-code.

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy.

Seaborn is a library for making statistical graphics in Python. It builds on top of matplotlib and integrates closely with pandas data structures. Seaborn helps you explore and understand your data.

NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object and tools for working with these arrays. It is the fundamental package for scientific computing with Python. Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using Numpy which allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use. The library is built

Jupyter notebook: The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations, and narrative text. It includes data cleaning and transformation, numerical simulation, statistical modelling, data visualization, machine learning.

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. It includes data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)
- Identification of possible problem-solving approaches (methods). The factors need to be found which can impact the used car price. This can be done by analysing the various factors and the stores the respondent prefers. This will be done by checking each of the factors impacts the respondents in decision making.
- Testing of Identified Approaches (Algorithms)
 - We utilized several classic and state-of-the-art methods, including ensemble learning techniques, with a 90% - 10% split for the training and test data. To reduce the time required for training, we used over 5 thousand examples from our dataset. Linear Regression, Random Forest and Gradient Boost were our baseline methods. For most of the model implementations, the open-source Scikit-Learn package was used.
 - **Linear Regression:** Linear Regression was chosen as the first model due to its simplicity and comparatively small training time. The features, without any feature mapping, were used

directly as the feature vectors. No regularization was used since the results clearly showed low variance.

- **Random Forest:** Random Forest is an ensemble learning based regression model. It uses a model called decision tree, specifically as the name suggests, multiple decision trees to generate the ensemble model which collectively produces a prediction. The benefit of this model is that the trees are produced in parallel and are relatively uncorrelated, thus producing good results as each tree is not prone to individual errors of other trees. This uncorrelated behaviour is partly ensured by the use of Bootstrap Aggregation or bagging providing the randomness required to produce robust and uncorrelated trees. This model was hence chosen to account for the large number of features in the dataset and compare a bagging technique with the following gradient boosting methods.
- **Decision Tree:** This is one of my favourite algorithm and I use it quite frequently. It is a type of supervised learning algorithm that is mostly used for the classification problems. Surprisingly it works for both categorical and continuous dependent variables. In this algorithm, we split the data into two or more homogeneous sets.
- **Gradient Boost:** Gradient Boosting is another decision tree based method that is generally described as “a method of transforming weak learners into strong learners”. This means that like a typical boosting method, observations are assigned different weights and based on certain metrics, the weights of difficult to predict observations are increased and then fed into another tree to be trained. In this case the metric is the gradient of the loss function. This model was chosen to account for non-linear relationships between the features and predicted price, by splitting the data into 100 regions.
- **KNN:** In order to capitalize on the linear regression results and the apparent categorical linearity in the data as indicated, an ensemble method which used KMeans clustering of the features and linear regression on each cluster was used. Due to large training time, a three-cluster model was used. Then, the dataset was classified into these three clusters and passed through a linear regressor trained on each of the three training sets
- **Run and Evaluate selected models**
- **Important package required**
- Our primary packages for this project are going to be pandas for data processing, NumPy to work with arrays, matplotlib & seaborn for data visualizations, and finally scikit-learn for building and evaluating our ML model.

Analysing the Data (EDA)

```
df.columns
```

```
Index(['index', 'Car Brand', 'Model', 'Price', 'Model Year', 'Location',  
      'Fuel', 'Driven (Kms)', 'Gear', 'Ownership', 'EMI (monthly)'],  
      dtype='object')
```

No. of Cars in different cities based on Ownership

```
location_owner = df.groupby(by=['Location', 'Ownership'])['Gear'].count().reset_index().rename(  
    columns={'Gear': 'Count'})  
location_owner
```

	Location	Ownership	Count
0	Bangalore	1	272
1	Bangalore	2	140
2	Bangalore	3	16
3	Bangalore	4	1
4	Chennai	1	489
5	Chennai	2	145
6	Chennai	3	20
7	Chennai	4	1
8	Delhi	1	612

Z-score

```
from scipy.stats import zscore

z_score = zscore(df[['Price', 'Driven (Kms)', 'EMI (monthly)']])
abs_zscore = np.abs(z_score)
filtering_entry = (abs_zscore < 3).all(axis=1)
df = df[filtering_entry]

# the data now seems much better than before.
df.describe()
```

	index	Car Brand	Model	Price	Model Year	Location	Fuel	Driven (Kms)	Gear	Ownership	EMI (monthly)
count	4680.000000	4680.000000	4680.000000	4.680000e+03	4680.000000	4680.000000	4680.000000	4680.000000	4680.000000	4680.000000	4680.000000
mean	2569.182479	11.583462	308.655983	5.540770e+05	2016.023932	4.023077	0.853832	47100.849573	0.867521	1.245299	12329.141867
std	1470.683107	4.464750	190.200228	2.085681e+05	2.203250	2.871098	0.487400	27810.949424	0.339047	0.489338	4800.899901
min	0.000000	0.000000	0.000000	1.203990e+05	2007.000000	0.000000	0.000000	58.000000	0.000000	1.000000	2678.000000
25%	1291.750000	7.000000	161.000000	4.039990e+05	2014.000000	2.000000	0.000000	25816.000000	1.000000	1.000000	8978.250000
50%	2554.500000	14.000000	295.500000	5.077990e+05	2016.000000	3.000000	1.000000	43298.500000	1.000000	1.000000	11297.000000
75%	3880.750000	14.000000	491.000000	6.645240e+05	2018.000000	7.000000	1.000000	65822.000000	1.000000	1.000000	14772.750000
max	5118.000000	21.000000	637.000000	1.273899e+06	2021.000000	8.000000	3.000000	130917.000000	1.000000	4.000000	28337.000000

Encoding:

Encoding For Categorical Variables

```
categorical_vars = df.select_dtypes(include=['object']).columns.tolist()
print("\n\nList of categorical features: \n", categorical_vars)
print("\n")
for x in categorical_vars:
    print("Distinct values in " + x + " : " + str(len(pd.unique(df[x]))) )
```

```
List of categorical features:
['Car Brand', 'Model', 'Location', 'Fuel', 'Gear']
```

```
Distinct values in Car Brand : 22
Distinct values in Model : 638
Distinct values in Location : 9
Distinct values in Fuel : 4
Distinct values in Gear : 2
```

Label Encoding for Brand, Car Name and city

```
from sklearn.preprocessing import LabelEncoder
num= LabelEncoder()

label_col = ['Car Brand', 'Model', 'Fuel', 'Gear', 'Location']
for x in label_col:
    df[x] = num.fit_transform(df[x].astype(str))

df.head()
```

	index	Car Brand	Model	Price	Model Year	Location	Fuel	Driven (Kms)	Gear	Ownership	EMI (monthly)
0	0	7	229	687499	2016	4	0	51885	1	1	15293
1	1	14	243	848899	2014	4	1	87232	1	1	14430
2	2	19	519	481999	2017	4	1	28702	1	2	10722
3	3	14	385	539999	2019	4	1	19377	1	1	11945
4	4	18	401	1292899	2017	4	1	47264	1	1	28760

VIF:

Calculate VIF Factors:

```
# For each X, calculate VIF and save in dataframe
vif = pd.DataFrame()
vif["VIF Factor"] = [variance_inflation_factor(df, x) for x in df.columns]
vif["features"] = x.columns
```

Inspect VIF Factors

```
vif.round(1)
```

	VIF Factor	features
0	4.6	index
1	7.8	Car Brand
2	3.9	Model
3	57.8	Model Year
4	3.8	Location
5	4.1	Fuel
6	5.1	Driven (Kms)
7	7.8	Gear
8	7.6	Ownership
9	8.5	EMI (monthly)

Compared to Linear Regression, most Decision-Tree based methods perform comparably well. This can be attributed to the apparent linearity of the dataset. We believe that It can also be attributed to the difficulty in tuning the hyper-parameters for most gradient boost methods

Model building :

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.tree import export_text

dt = ExtraTreesRegressor()
dt.fit(x_train,y_train)

y_pred = dt.predict(x_test)

print("Adjusted R2 squared : ",dt.score(x_train,y_train))
print("Mean Absolute Error (MAE): ", mean_absolute_error(y_test, y_pred))
print("Mean Squared Error (MSE): ",mean_squared_error(y_test, y_pred))
print("Root Mean Squared Error (RMSE): ",np.sqrt(mean_squared_error(y_test, y_pred)))

Adjusted R2 squared : 1.0
Mean Absolute Error (MAE): 38676.33172080166
Mean Squared Error (MSE): 12973197230.61783
Root Mean Squared Error (RMSE): 113899.9439447528
```

```
from sklearn.ensemble import RandomForestRegressor

rf = RandomForestRegressor()
rf.fit(x_train,y_train)

y_pred = rf.predict(x_test)

print("Adjusted R2 squared : ",rf.score(x_train,y_train))
print("Mean Absolute Error (MAE): ", mean_absolute_error(y_test, y_pred))
print("Mean Squared Error (MSE): ",mean_squared_error(y_test, y_pred))
print("Root Mean Squared Error (RMSE): ",np.sqrt(mean_squared_error(y_test, y_pred)))

Adjusted R2 squared : 0.9724954999291008
Mean Absolute Error (MAE): 37086.61506565307
Mean Squared Error (MSE): 12163001828.04423
Root Mean Squared Error (RMSE): 110286.00014527787
```

```
# ExtraTreesRegressor

from sklearn.ensemble import ExtraTreesRegressor

ext_reg = ExtraTreesRegressor()
ext_reg.fit(x_train,y_train)

y_pred = ext_reg.predict(x_test)

print("Adjusted R2 squared : ",ext_reg.score(x_train, y_train))
print("Mean Absolute Error (MAE): ", mean_absolute_error(y_test, y_pred))
print("Mean Squared Error (MSE): ",mean_squared_error(y_test, y_pred))
print("Root Mean Squared Error (RMSE): ",np.sqrt(mean_squared_error(y_test, y_pred)))

Adjusted R2 squared : 1.0
Mean Absolute Error (MAE): 38094.14789219074
Mean Squared Error (MSE): 12673508903.208017
Root Mean Squared Error (RMSE): 112576.68010386528
```

```
# KNN

from sklearn.neighbors import KNeighborsRegressor

k_neigh = KNeighborsRegressor()
k_neigh.fit(x_train,y_train)

y_pred = k_neigh.predict(x_test)

print("Adjusted R2 squared : ",k_neigh.score(x_train,y_train))
print("Mean Absolute Error (MAE): ", mean_absolute_error(y_test, y_pred))
print("Mean Squared Error (MSE): ",mean_squared_error(y_test, y_pred))
print("Root Mean Squared Error (RMSE): ",np.sqrt(mean_squared_error(y_test, y_pred)))

Adjusted R2 squared : 0.7771330609010983
Mean Absolute Error (MAE): 87249.12232204562
Mean Squared Error (MSE): 22093449627.09053
Root Mean Squared Error (RMSE): 148638.65455220768
```

```
from sklearn.ensemble import GradientBoostingRegressor

grid_reg = GradientBoostingRegressor()
grid_reg.fit(x_train,y_train)

y_pred = grid_reg.predict(x_test)

print("Adjusted R2 squared : ",grid_reg.score(x_train,y_train))
print("Mean Absolute Error (MAE): ", mean_absolute_error(y_test, y_pred))
print("Mean Squared Error (MSE): ",mean_squared_error(y_test, y_pred))
print("Root Mean Squared Error (RMSE): ",np.sqrt(mean_squared_error(y_test, y_pred)))

Adjusted R2 squared : 0.8766242338517425
Mean Absolute Error (MAE): 39861.53188529727
Mean Squared Error (MSE): 10349634170.092371
Root Mean Squared Error (RMSE): 101733.15177508447
```

```
lr = LinearRegression()
lr.fit(x_train,y_train)

y_pred = lr.predict(x_test)

print("Adjusted R2 squared : ",lr.score(x_train,y_train))
print("Mean Absolute Error (MAE): ", mean_absolute_error(y_test, y_pred))
print("Mean Squared Error (MSE): ",mean_squared_error(y_test, y_pred))
print("Root Mean Squared Error (RMSE): ",np.sqrt(mean_squared_error(y_test, y_pred)))

Adjusted R2 squared : 0.5767838746757535
Mean Absolute Error (MAE): 80606.47776446874
Mean Squared Error (MSE): 22414746885.919395
Root Mean Squared Error (RMSE): 149715.5532532255
```

- Key Metrics for success in solving problem under consideration
- Using the sklearn.metrics I have calculated Adjusted R2 squared ,Mean Absolute Error (MAE),Mean Squared Error (MSE),Root Mean Squared Error (RMSE)
- Using Hyper-parameter : model parameters are estimated from data automatically and model hyper-parameters are set manually and are used in processes to help estimate model and Grid search is a basic method for hyper-parameter tuning. It performs an exhaustive search on the hyper-parameter set specified by users.

Hyper-Parameter tuning

```
from sklearn.model_selection import GridSearchCV
from pprint import pprint
pprint(grid_reg.get_params())

{'bootstrap': True,
 'ccp_alpha': 0.0,
 'criterion': 'mse',
 'max_depth': None,
 'max_features': 'auto',
 'max_leaf_nodes': None,
 'max_samples': None,
 'min_impurity_decrease': 0.0,
 'min_impurity_split': None,
 'min_samples_leaf': 1,
 'min_samples_split': 2,
 'min_weight_fraction_leaf': 0.0,
 'n_estimators': 100,
 'n_jobs': None,
 'oob_score': False,
 'random_state': None,
 'verbose': 0,
 'warm_start': False}
```



- Using cross-validation for evaluating ML models by training several ML models on subsets of the available input data and evaluating on the complementary subset of the data. Use cross-validation to detect overfitting,

Cross Validation

```
from sklearn.model_selection import cross_val_score

scr = cross_val_score(dt, x, y, cv=5)
print("Cross Validation score of DecisionTreeRegressor model is:", scr.mean())

scr = cross_val_score(rf, x, y, cv=5)
print("Cross Validation score of RandomForestRegressor model is:", scr.mean())

scr = cross_val_score(ext_reg, x, y, cv=5)
print("Cross Validation score of ExtraTreesRegressor model is:", scr.mean())

scr = cross_val_score(k_neigh, x, y, cv=5)
print("Cross Validation score of KNeighborsRegressor model is:", scr.mean())

scr = cross_val_score(lr, x, y, cv=5)
print("Cross Validation score of LinearRegression model is:", scr.mean())

scr = cross_val_score(grid_reg, x, y, cv=5)
print("Cross Validation score of GradientBoostingRegressor model is:", scr.mean())

Cross Validation score of DecisionTreeRegressor model is: 0.6410050065541608
Cross Validation score of RandomForestRegressor model is: 0.6541615245644753
Cross Validation score of ExtraTreesRegressor model is: 0.6446531796860746
Cross Validation score of KNeighborsRegressor model is: 0.44014264573184436
Cross Validation score of LinearRegression model is: 0.5085059198723307
Cross Validation score of GradientBoostingRegressor model is: 0.6649131849500962
```

Visualizations

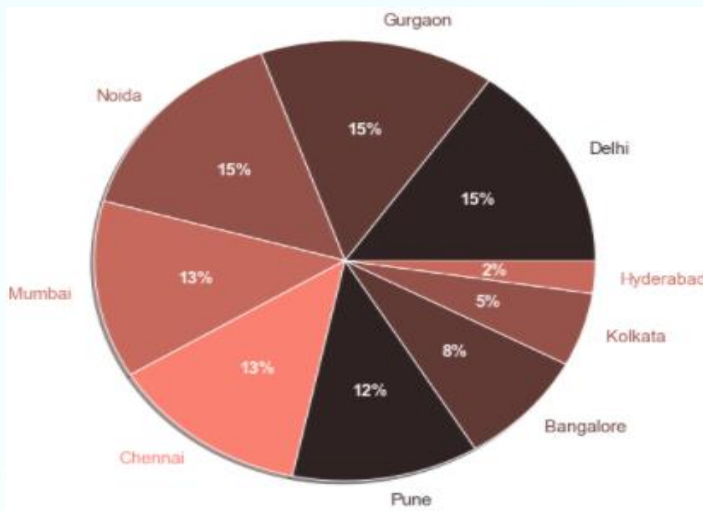
Mention all the plots made along with their pictures and what were the inferences and observations obtained:

Visualizing the Data

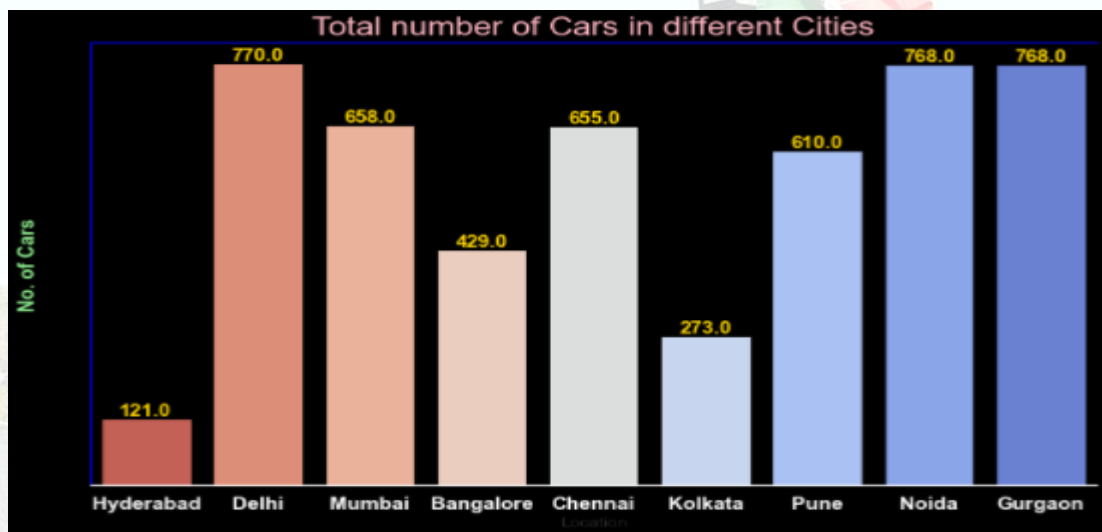
```
df.Location.unique()

array(['Hyderabad', 'Delhi', 'Mumbai', 'Bangalore', 'Chennai', 'Kolkata',
      'Pune', 'Noida', 'Gurgaon'], dtype=object)
```

Percentage of cars available for sale in different cities:

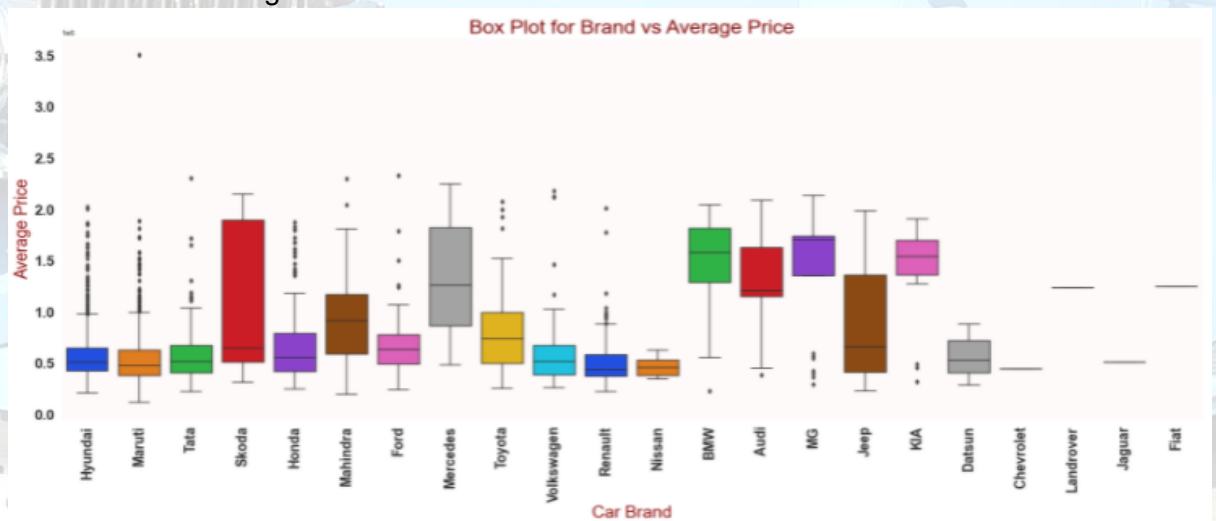


Total number of cars in different Cities

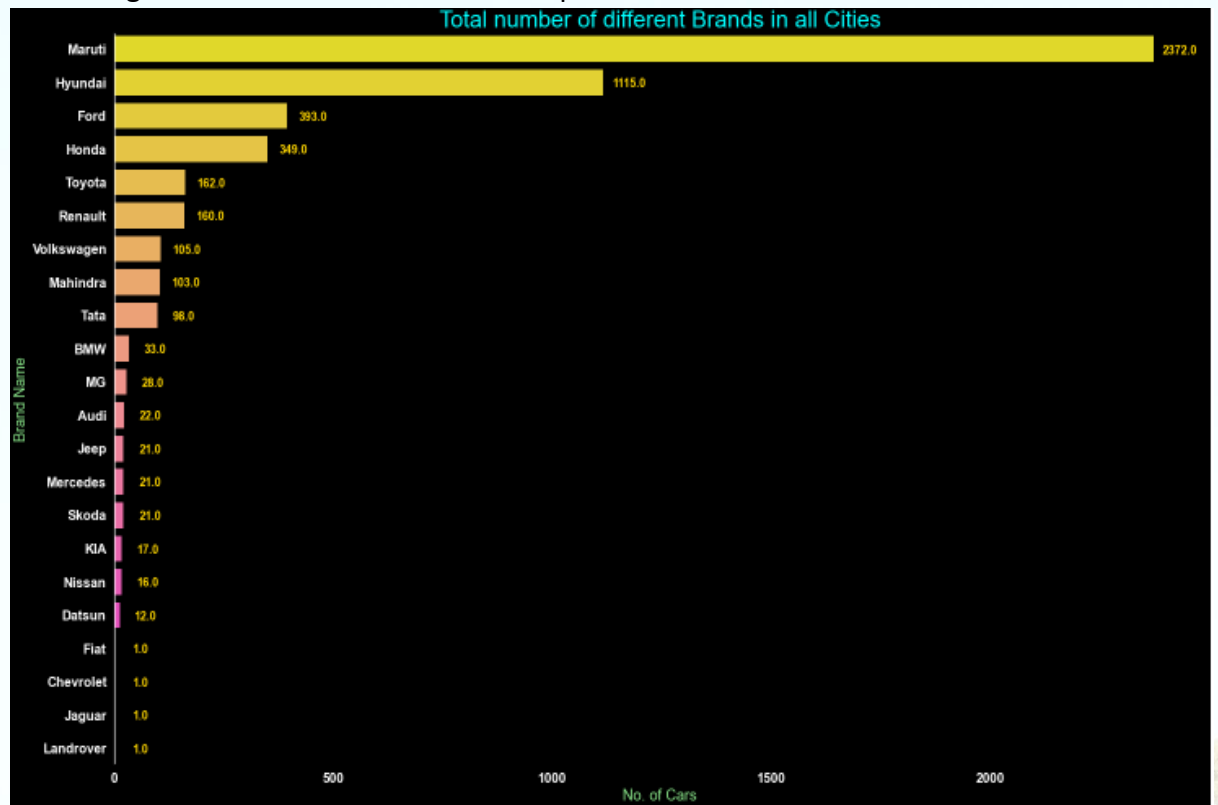


No. of different Brands in all Cities:

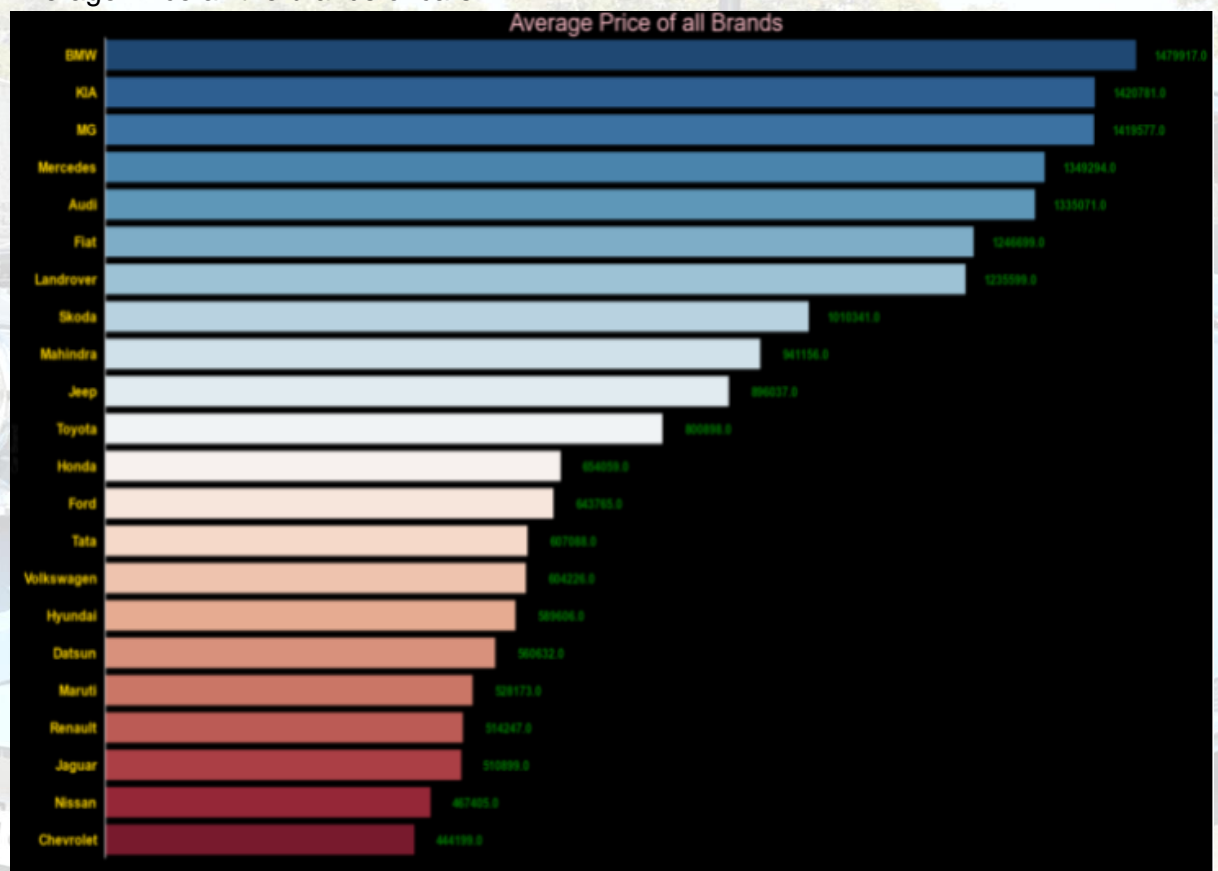
Car Brand vs Average Price:



Percentage of Maruti car are more as compare to other.

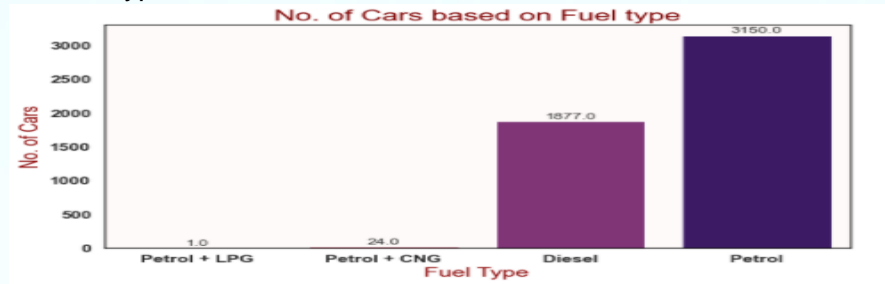


Average Price all the brands of cars:

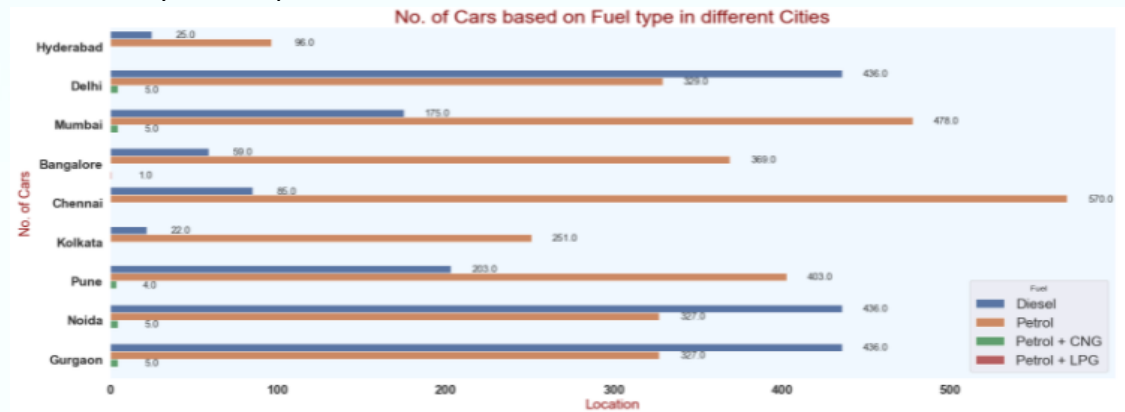


Number of Cars based on Fuel type in all cities:

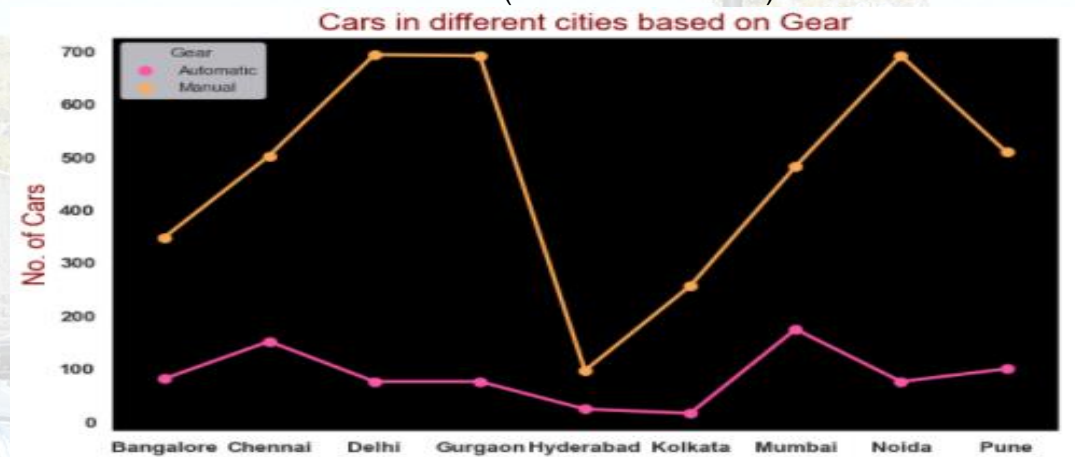
	Fuel	Count
0	Petrol + LPG	1
1	Petrol + CNG	24
2	Diesel	1877
3	Petrol	3150



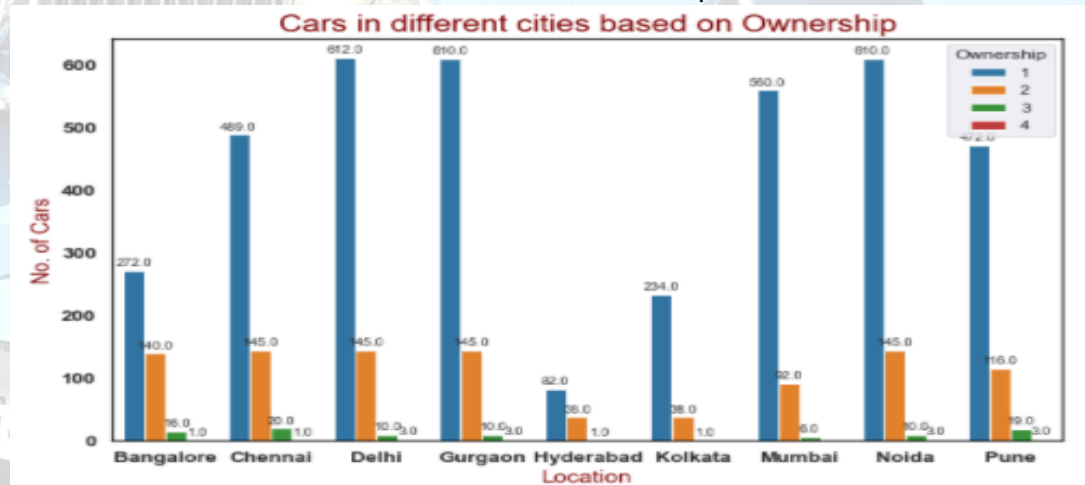
Fuel used by cars as per locations:



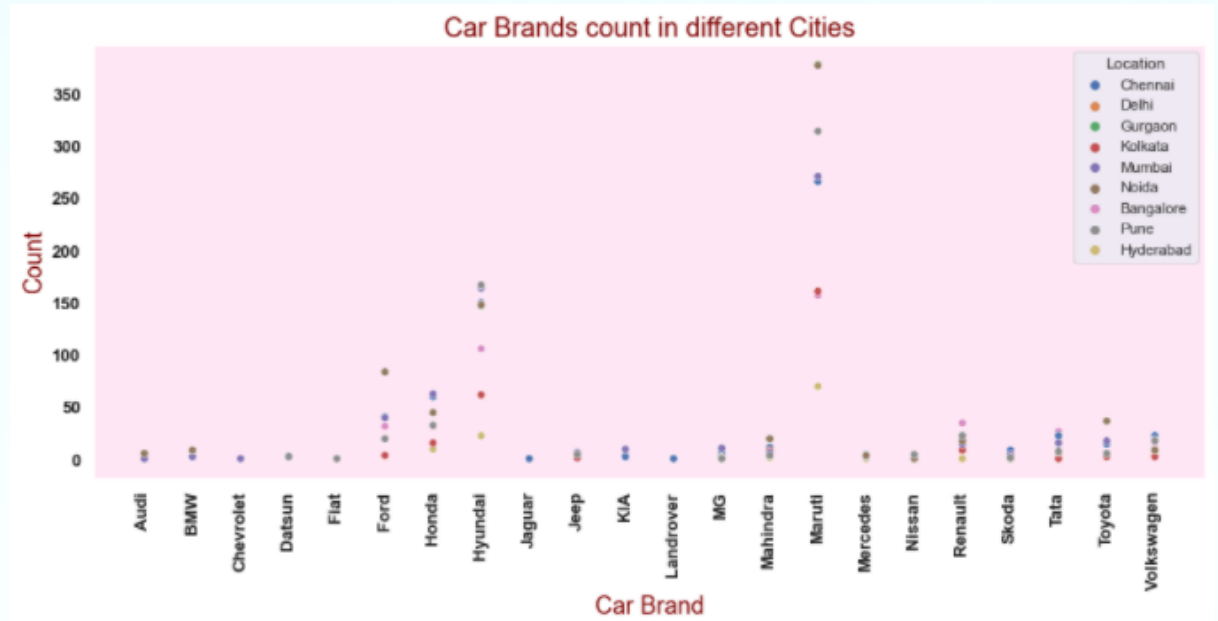
Cars in different cities based on Gear (i.e: Automatic / Manual)



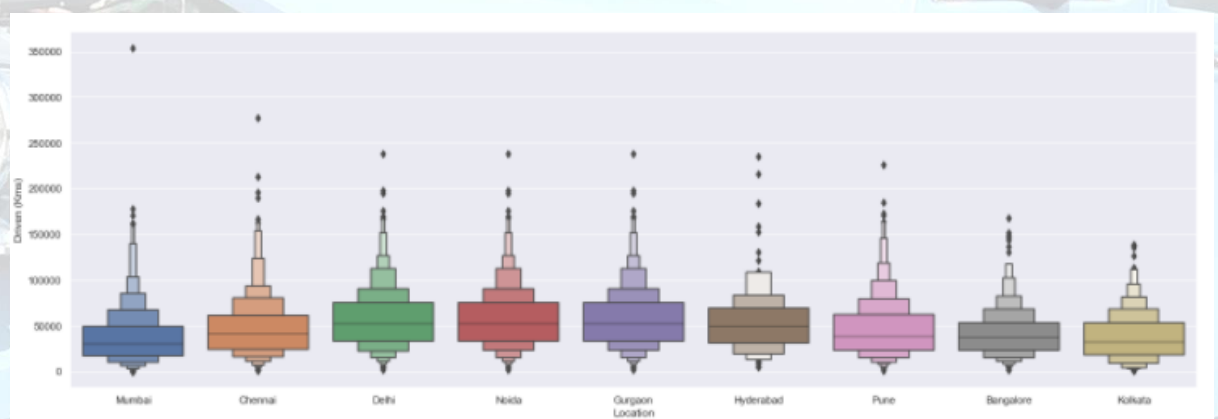
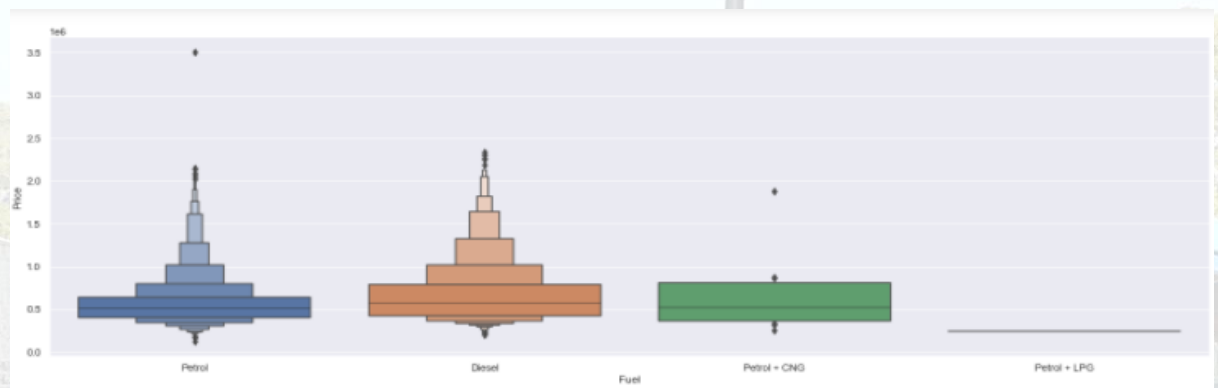
Number of Cars in different cities based on Ownership:



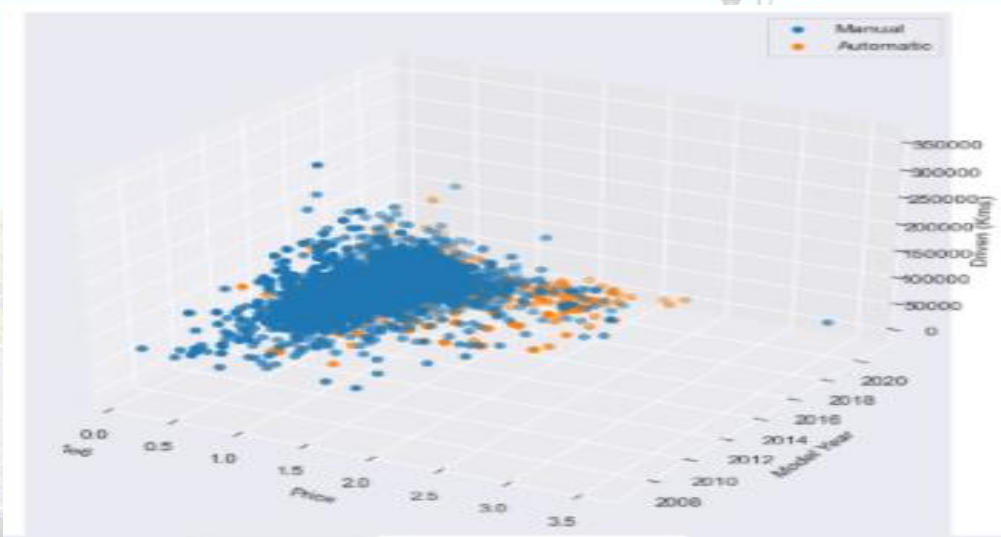
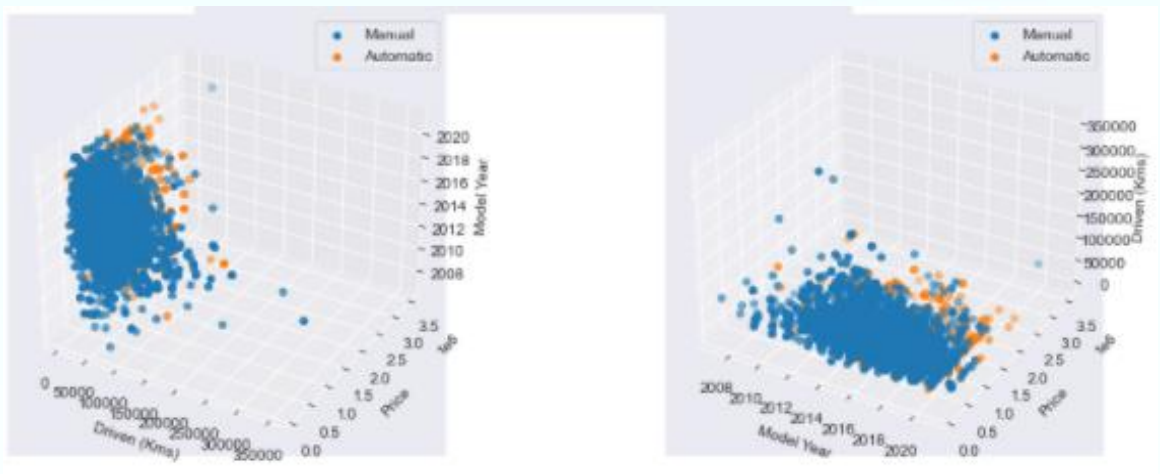
Relation of car brand count in different Cities:



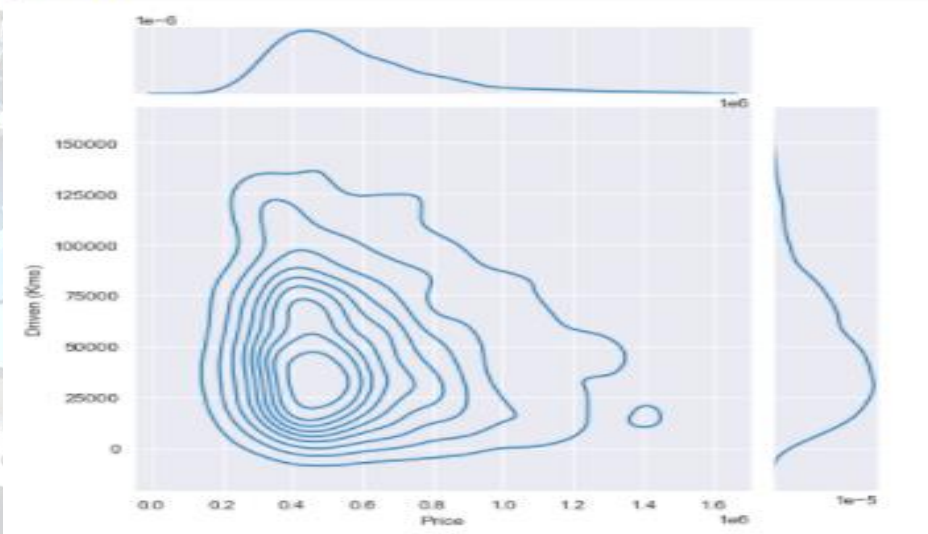
Relation between Average Price, EMI, Driven(Kms), location, Fuel type:



3D Visualizations:



```
sns.set_style("darkgrid")
sns.jointplot("Price", "Driven (Kms)", data=df, kind="kde")
plt.show()
```



● Interpretation of the Results

- In this research, two experiments were performed, the first experiment was collecting data using all the variables available in the dataset after pre-processing, while the second experiment was conducted using most important variables and the goal of this is to be able to improve the model's performance using fewer variables.
- There requirement of train and test and building of many models to get accuracy of the model.
- There are multiple of matric which decide the best fit model like as : R-squared ,RMSE value, VIF, CDF & PDF Z-score and etc.
- Database helped in making perfect model and will help in understanding Indian market of used Cars.

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. This paper compares different algorithms for machine learning.
- Future Analysis : 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 and can filter or add data as per the costumer's requirements.
- India will be the biggest market for the used cars, used car market in India has been the center of attention in the slow growing automotive industry in India. In the last year, demand for used cars has soared with over 42 lakh buyers (Source: CRISIL).
- The report in "The Indian Express" also quotes a study by the consulting firm Mckinsey, that indicates continued growth in the used-car segment with India projected to be the 3rd largest market for used-cars by 2021.

● Learning Outcomes of the Study in respect of Data Science

- Any aspiring young buyer is likely to go for a second-hand car like a used Maruti WagonR or a used Alto instead of a new cars.
- The trend in the used car market in terms of buyer preference and requirements mimic the trend in the new car market. However, buyers are more likely to experiment with brands in the used car segment. It is not the case with new cars. Maruti Suzuki though has always been

the top choice of used car buyers and also enjoys a majority market share in the new car market.

- Report says that even though India has seen immense growth in the used car market, there is still potential for future growth through the organized sector since in mature markets like the US and Europe the ratio of new cars to old cars stand at 1:3. All in all, the Indian used car market seems to be heading in the right direction.
- **Limitations of this work and Scope for Future Work**
 - About used car industry: It is the huge industry and the data given is quite small but for getting a start and for the decision making the data is quite sufficient. In this industry there is always open opportunities to get start. There are different raw data are available for real estate and applying data science to it, will move this industry at a next level.
 - We can get the understanding of past, present and the future market of used cars. Today in fast moving world this could be a great investment for business. To beat the market there is requirement to understanding the value of data science in the field of real estate. Data can be driven and it will open opportunity for the costumer as well as the seller.

