Final Lab on Unsupervised learning using Customer Personality analysis dataset

Main Objective: In today's market with inflation happening in almost every sectors of our life it is very important to be aware of how much we are willing to spend on expensive buying. Buying a car is an expensive aspect of our lives, since it is something that we are going to use for the next 5 years of our life. In that aspect it is important to be aware on how much we are willing to spend on it such that we are able to make sure it is a good buy.

In this dataset we have information about used cars and main objective is to be able to predict the selling price for these cars. As described in the above section we have certain information about the car using which we will be trying to come up with a deep neural network model that would be able to predict the car price.

Here we will be trying at least 3 approaches

* Try predicting the car prices using a simple neural network model, probably with one hidden layer

* Next I will be trying with a deep neural network maybe with at least 3 hidden layers

* Finally I will see if using some sort of penalty helps with the prediction

Description about the data: I found this data from Kaggle, it is a dataset containing information about cars (Used plus new). The idea behind the data set is to be able to predict the car selling price.

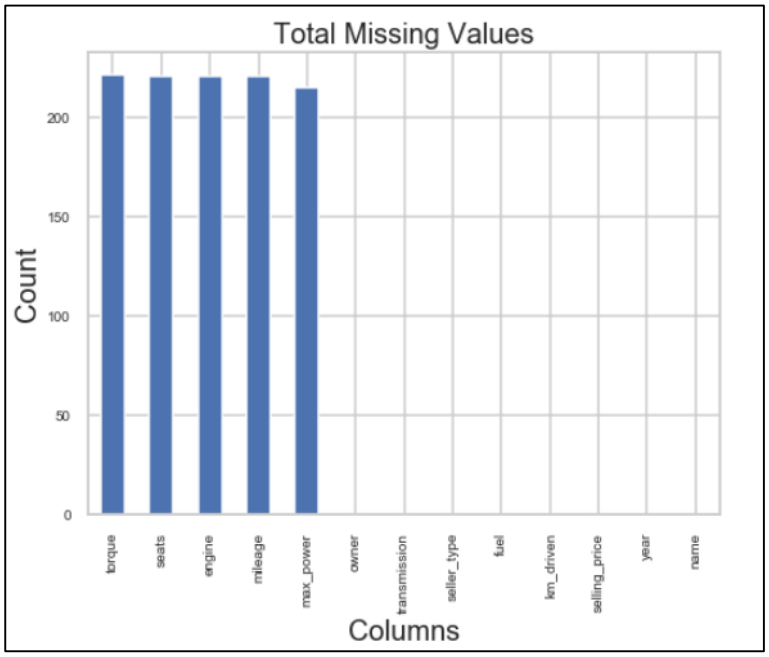
Following are the features in the data set and a few descriptions about the data

- name: categorical feature; Name of the cars ex. Maruti Suzuki, Hyundai Verna etc.
- year: categorical feature; Year when the car was manufactured
- selling_price: continuous feature; Price at which the car is offered to be sold, this is the predictor variable
- km_driven: continuous feature; Number of kilometers the car has been driven
- fuel: categorical feature; Fuel type of the car ex. Petrol, diesel, cng, lpg, electric
- seller_type: categorical feature; Describes if the seller is an individual or a dealership
- transmission: categorical feature; Gear trasmission of the car automatic or Manual
- Owner: categorical feature; tells if the owner is the first owner or second etc.
- mileage: mileage of the car
- engine: Engine of the car
- max power: Max power
- torque:
- seats: 4,5 etc.

Data Exploration and Feature Engineering Summary:

Handling Missing Values:

In the dataset few columns had less than 2% missing values, these were mainly torque, seats, engine, mileage and max power. For these rows since the amount of missing values were pretty small I removed those rows from the data frame.



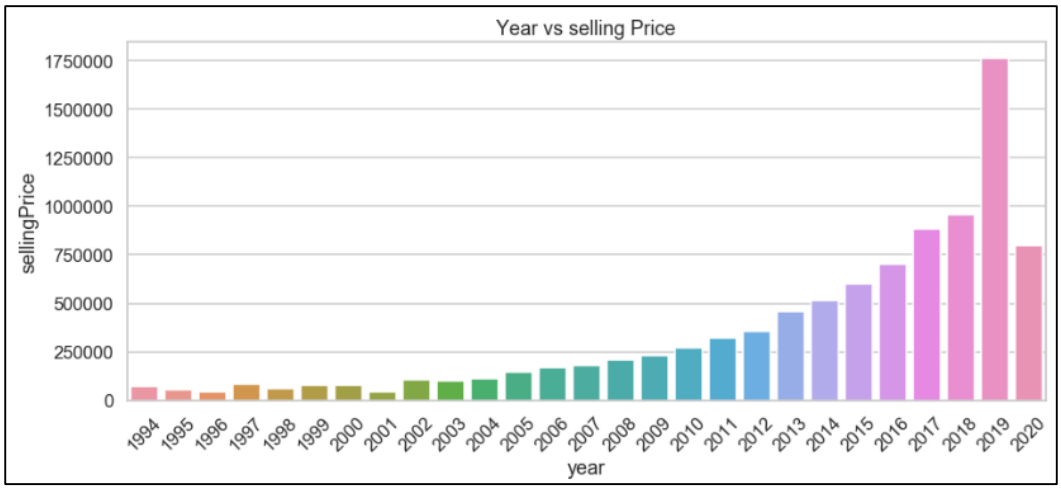
Exploratory Data Analysis:

In this section I went by each of the variables available in the dataset and tried to look into the values and derive information from the variable by transforming the variables

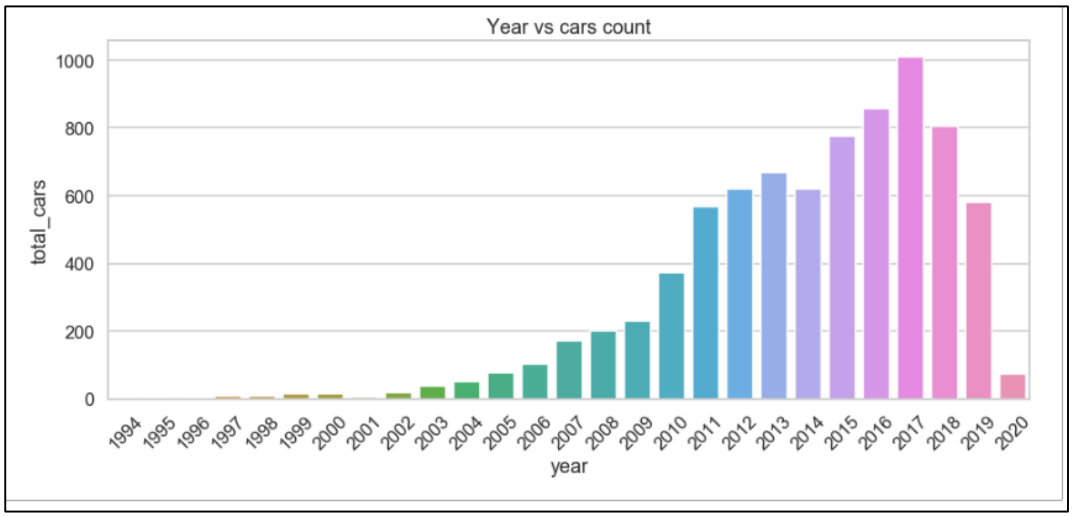
In this dataset we had around 10 categorical variables and 2 continuous variables.

Let's look at these variables one by one

Year – this variable mainly talks about the year since the car was manufactured, using this variable I looked into the average car price based on the year of manufacture

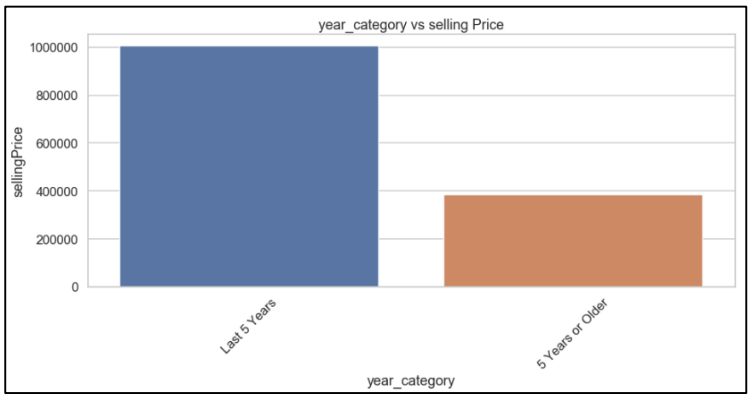


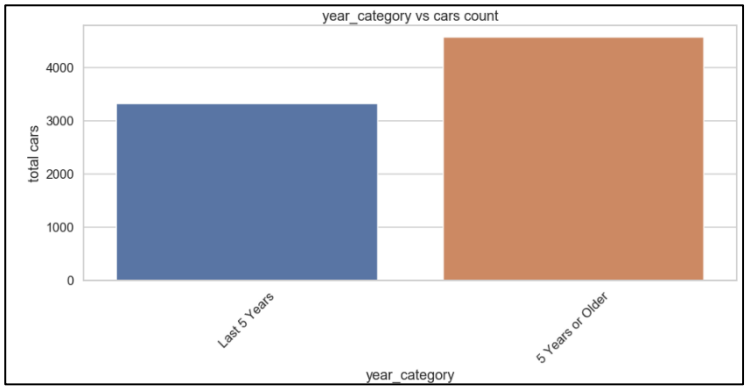
It can be observed there is a high relation between car selling prices and number of years used with higher the number of years the car has been used lower is the price. It can also be observed that there are no brand new cars in this dataset. Also car prices were higher in 2019 compared to 2020.



In the above plot I am checking the number of cars from each of these years and it can be see there are very few antique cars and most of the cars are from 2010 onwards.

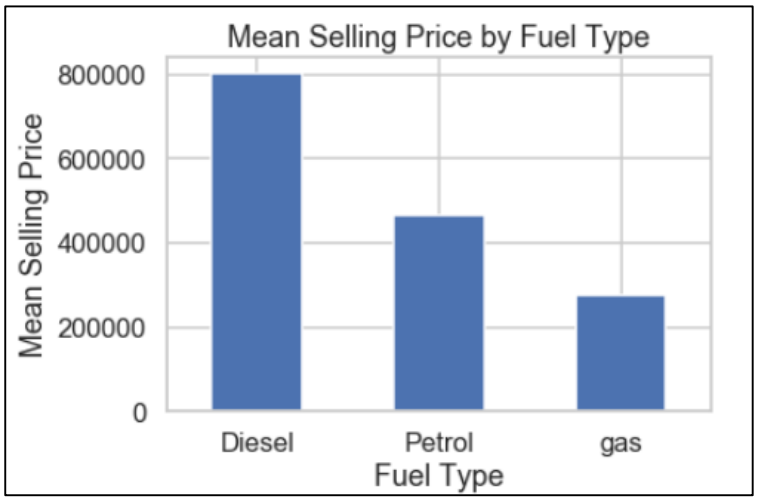
Using this knowledge, I divided the year category into two categories less than 5 years, i.e 2015 to 2020 and greater than 5 years i.e. 2015 and before

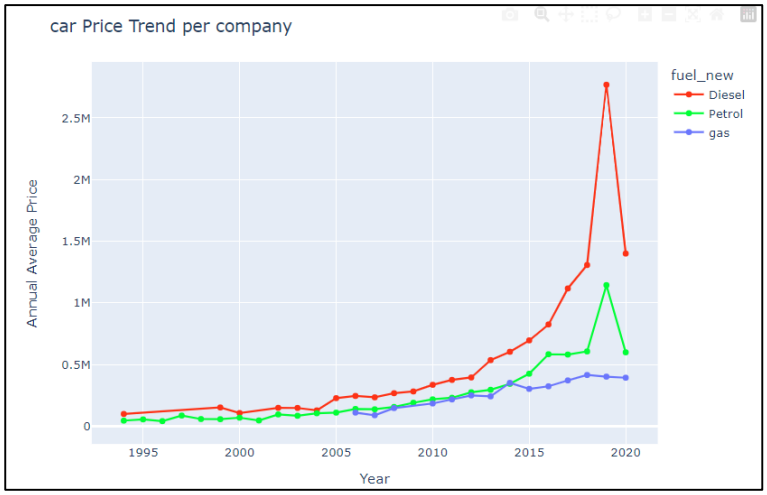




It can be concluded that newer cars have a significantly higher prices

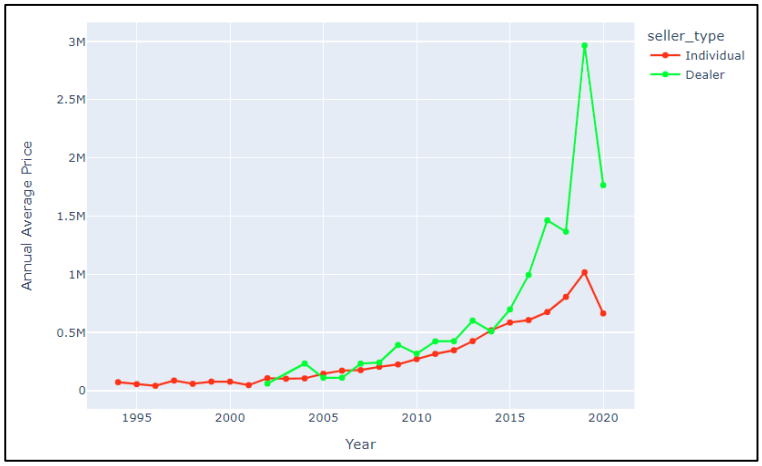
Fuel_Type: In the dataset there were mainly 4 different type of fuel cars namely Diesel, Petrol, CNG and LPG. Overall oil cars were much more than gas cars. I merged CNG and LPG categories into one category gas and kept the other Petrol and Diesel as it is.





Diesel car prices have always been a slightly higher than Petrol car prices with recently the price difference from 2012 there is a significant difference

Seller Type: Seller type variable tells us about who is selling the car, in this dataset there were mainly three categories mainly Dealer, Trustmark Dealer and Individual. Since Trustmark Dealer is also a type of dealer I merged them into one category.

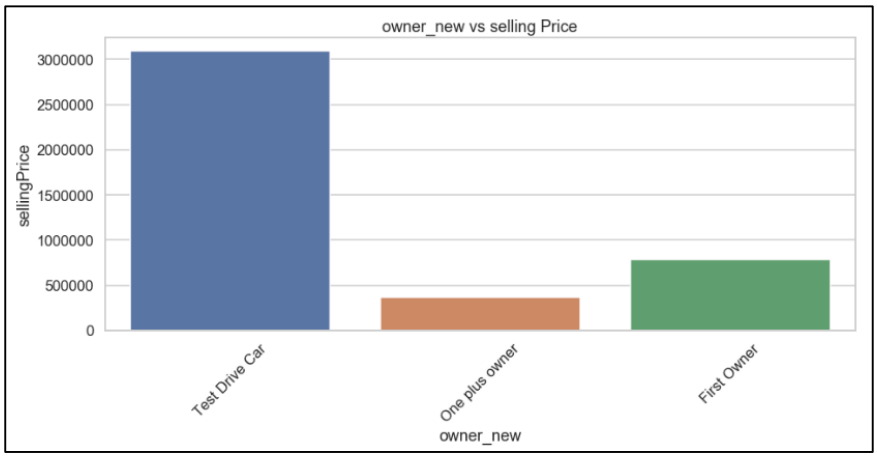


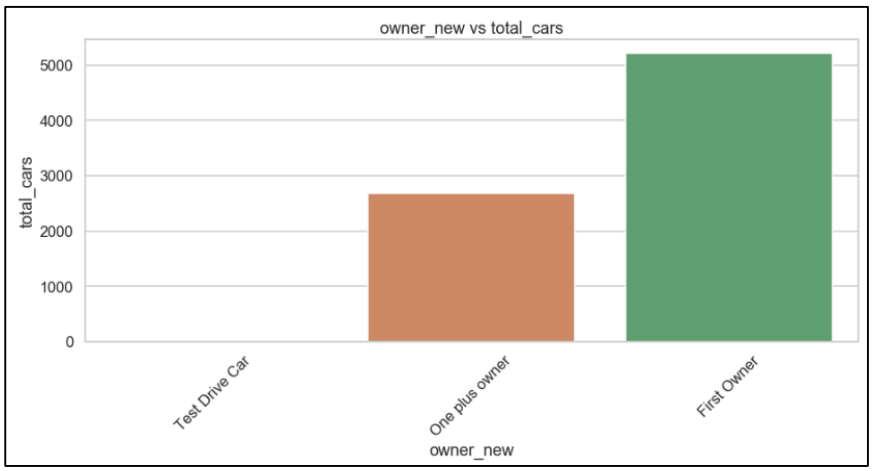
Selling Price of cars by Dealership started showing up from around 2002 and was in similar line as that of Individual sellers but we saw a huge spike from around 2015, with the onset of many new esteemed dealerships.

Owner: Tells us about the current owner of the car, it has 5 categories test drive car symbolizing brand new car, First, Second, Third owner etc. As it can be assumed new cars will have the most prices and the price detoriates as the number to times the car ownership has changed. People usually don't like to buy cars that have been owned by more than one people.

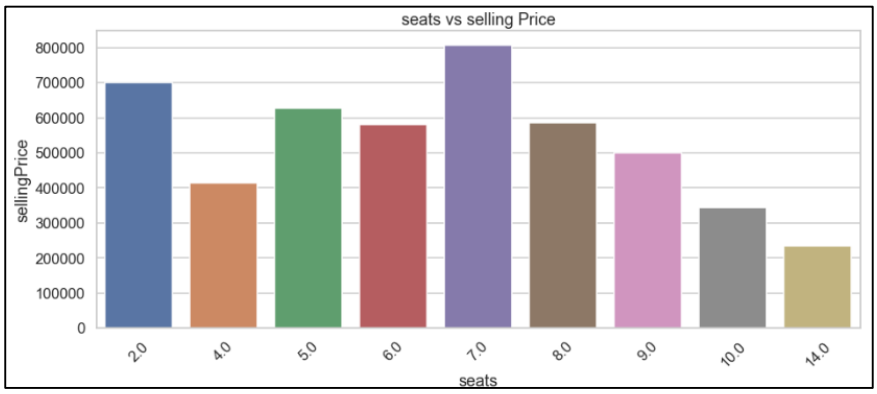
	owner	avg_selling_prices	total_cars
0	Third Owner	292977.0	509
1	Test Drive Car	3091000.0	3
2	Second Owner	401240.0	2010
3	Fourth & Above Owner	233197.0	160
4	First Owner	787051.0	5209

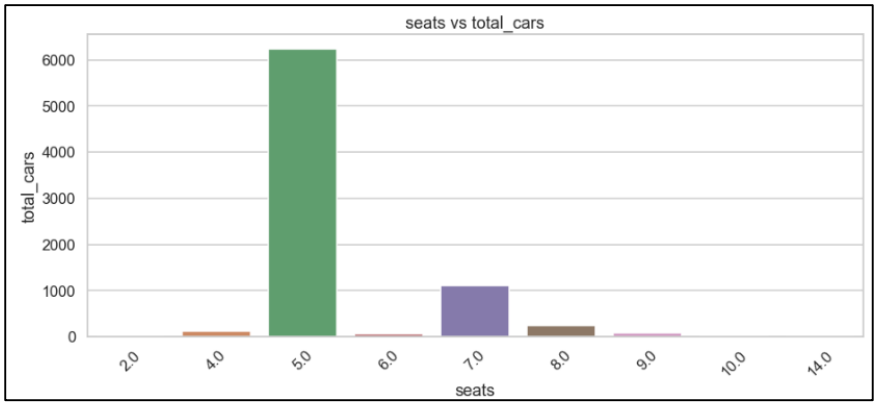
There are only 3 cars that are test drive cars in the dataset and price wise it is much more than the other cars whereas most of the cars are at least had one owner and rest are more divided. Keeping in mind new cars and cars owned only once will have significant prices difference from cars owned more than once I created three categories here.





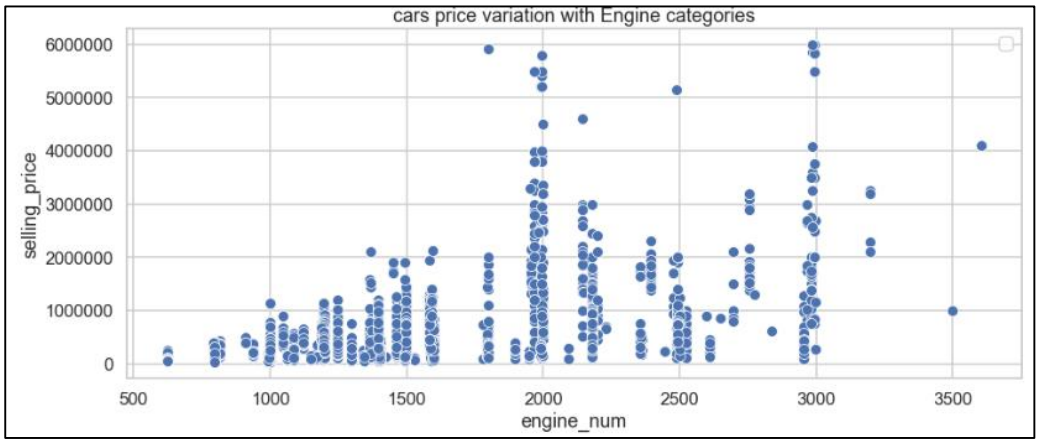
Seats: Variable seats talks about number of seats in the car



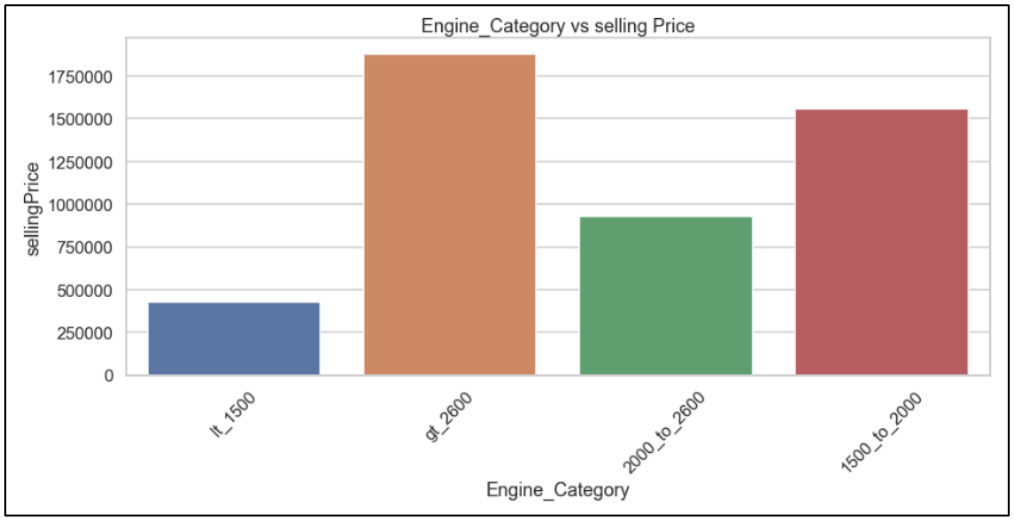


7 seater cars have the most price followed by 2 seater cars. Which defines two complexly different kind of cars. Where 2 seaters are mostly sports version and 7 seaters are family cars. But the dataset mostly has 5 seater cars and 7 seater cars rest of them have very few cars. I reduced the categories into less than 5 and more than 5 categories.

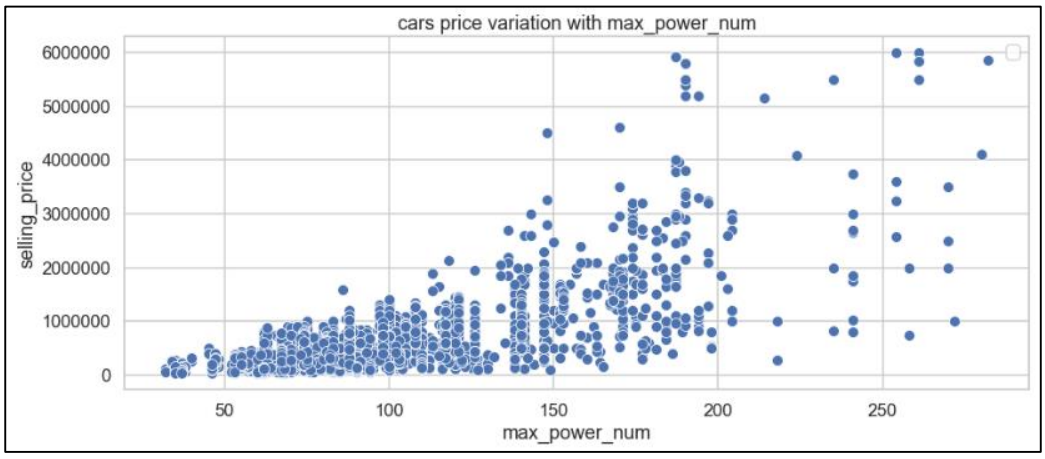
Engine: There are cars with varying type of engines. The engine variable was an object type variable with numerical string numbers. I first converted them into numeric and visualized them



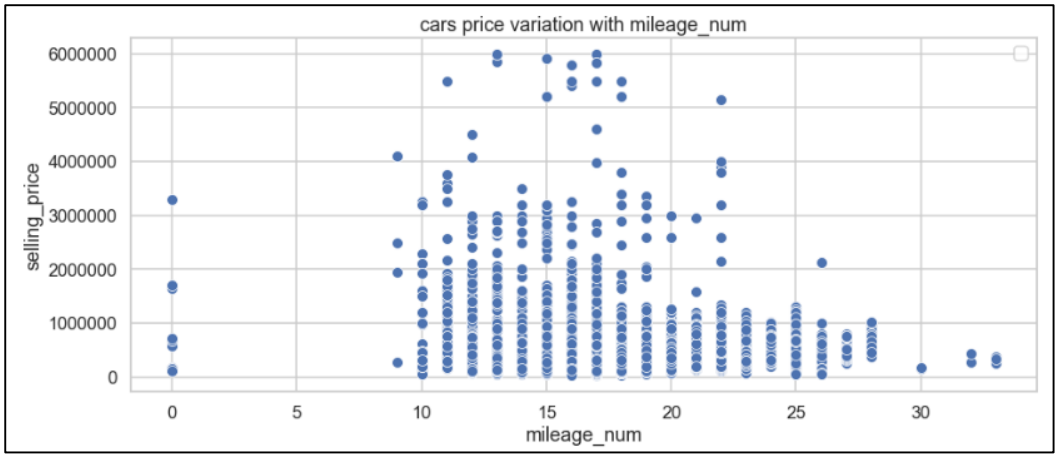
From the above plot we can see that engine size ranges from 500 cc to 3500 cc and also we can see that there clearly are some categories. I divided them into smaller buckets based on the cc and the selling prices. Mainly I created 4 categories less than 1500, 1500 to 2000 cc, 2000cc to 2600 cc and 2600cc and above.



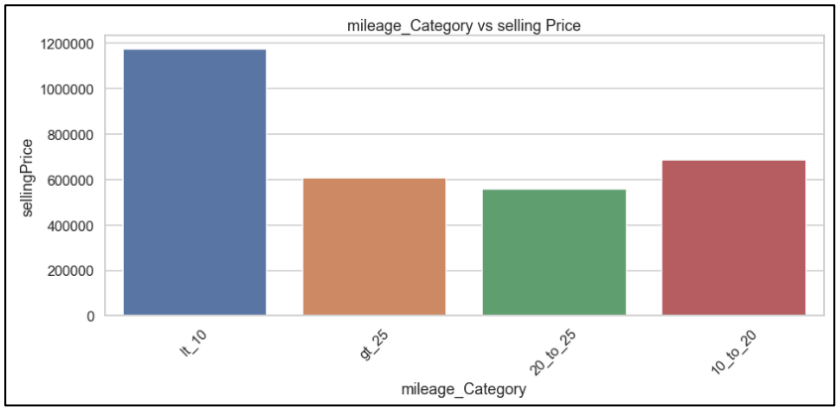
Max_Power: Maximum power of the car, on transforming the numerical string values to numerical I found there is a strong liner relationship between Max power and selling price

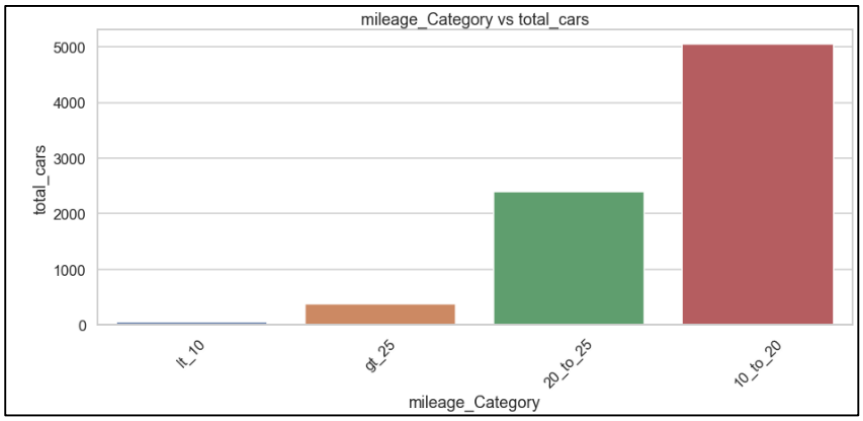


Mileage: Again similar to Engine and Max power I first converted this column into integer values and then plot it against Selling price to check for relationship



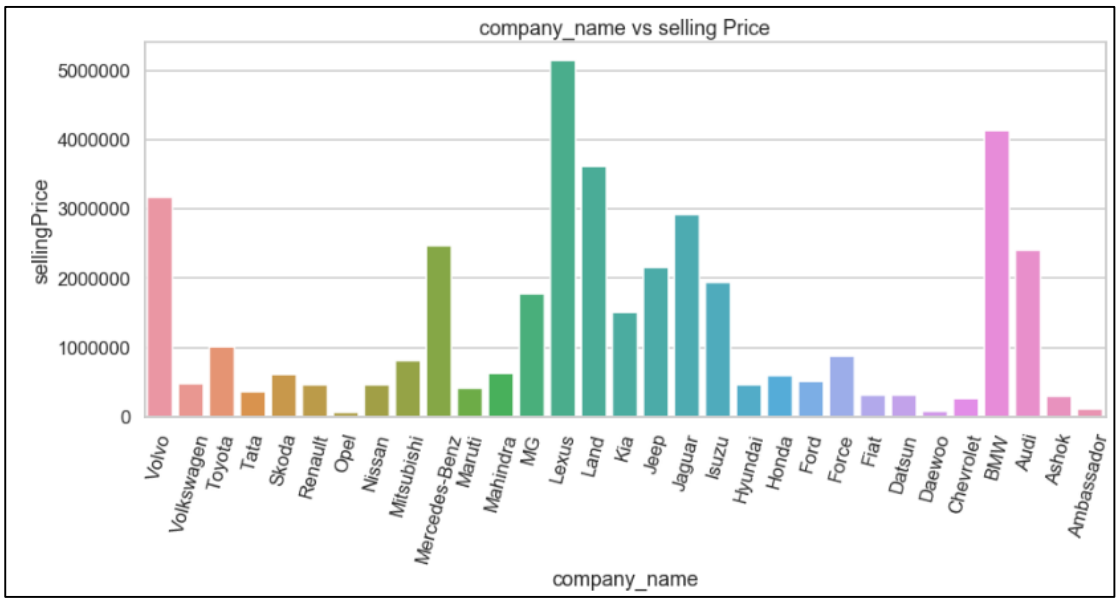
I do not see a clear relationship but there are few cars with 0 mileage and then cars in the range of 10 to 20 mileage have the most selling price and then it drops again in around 25 and 30. Using this I created 4 categories less than 10, 10 to 20, 20 to 25 and 25 above.

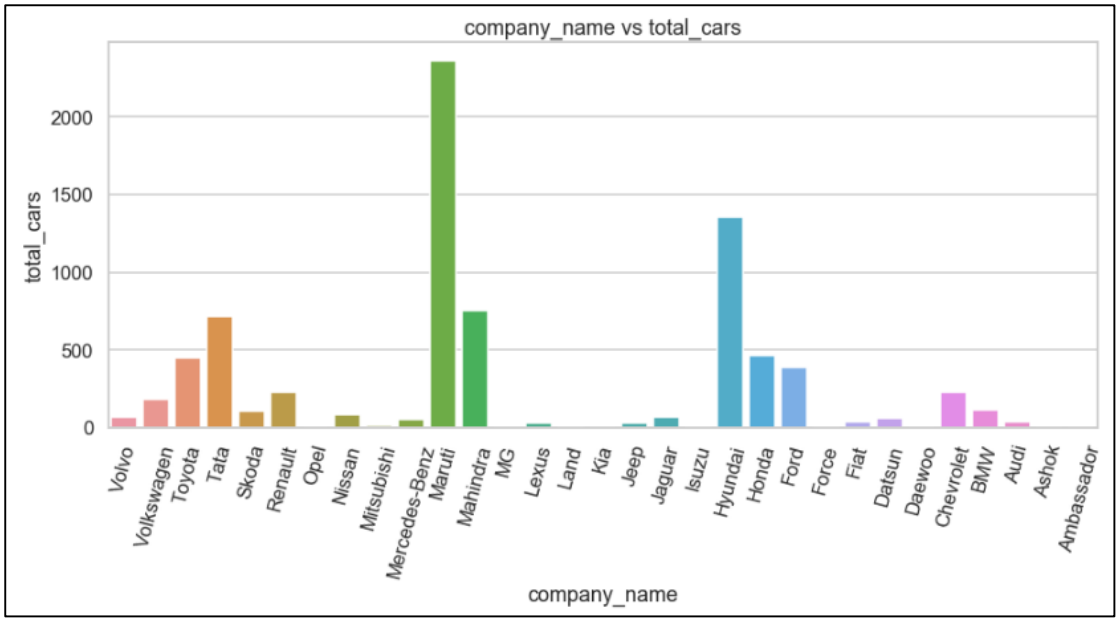




With this division I can clearly say that cars with mileage less than 10 have quite high prices, this can be because these are newer cars followed by the other categories. But most of the cars are in the range of 10 to 25.

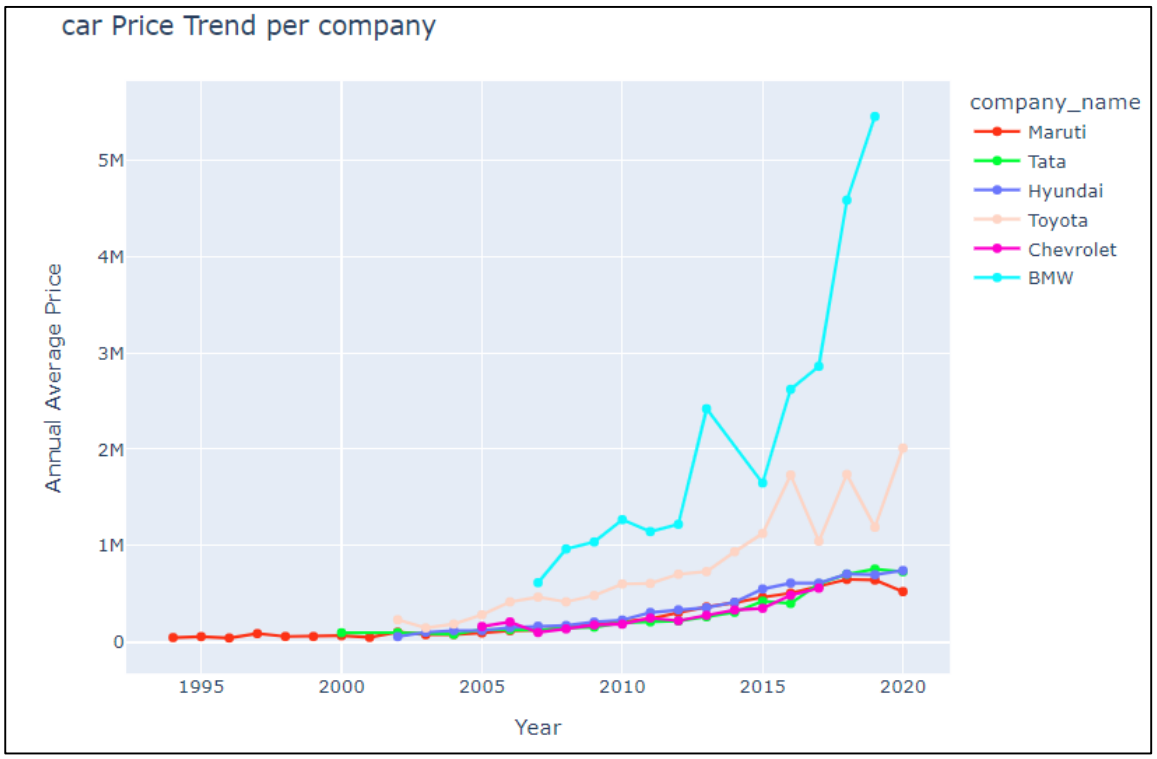
Name: Next using the car names I created some derived features. I was able to determine the car manufacturing companies and from there I was able to determine the country from which the car was created as well as the type of car i.e. Luxury, Premium Economy.



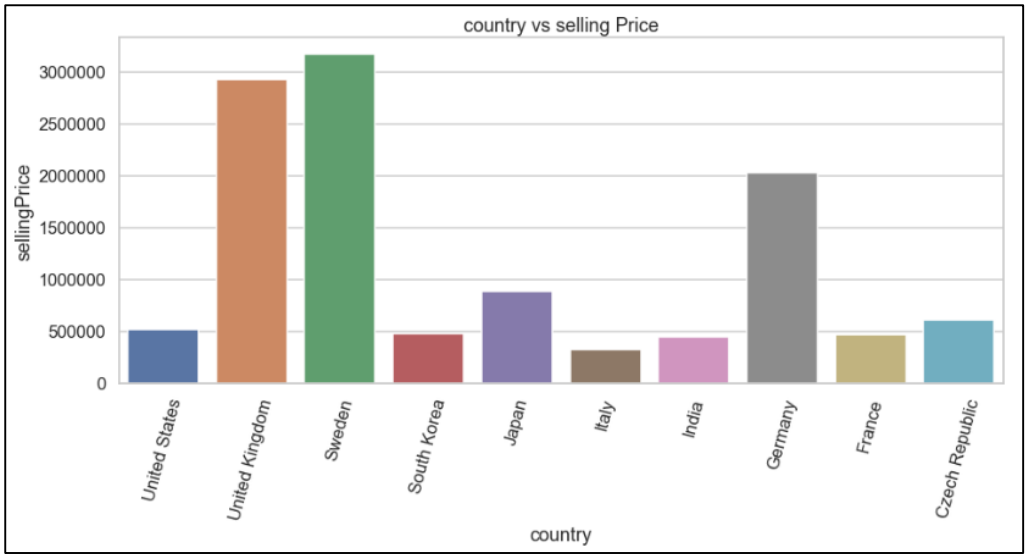


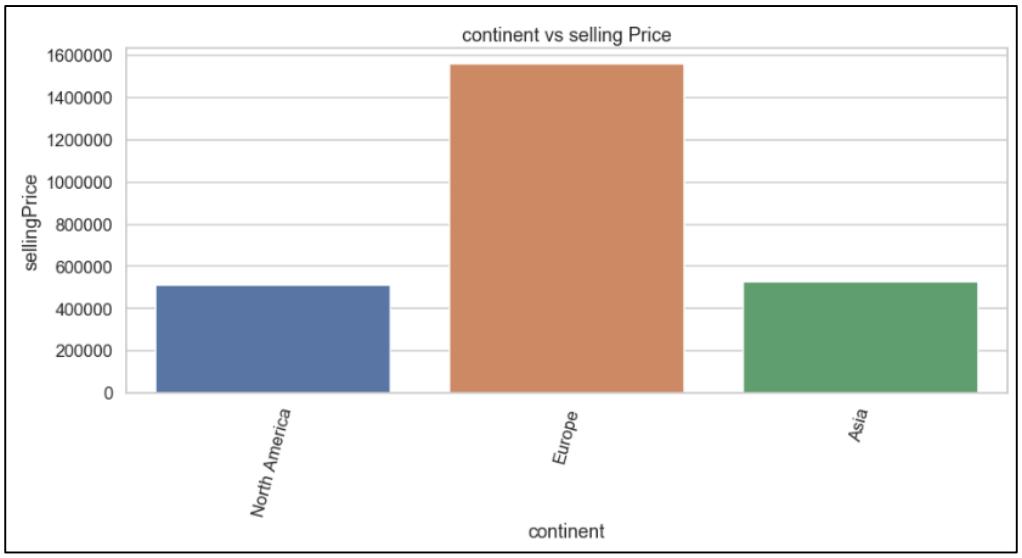
companies with the most cars are mainly Maruti, Mahindra Hyundai, Tata - these are mostly cars predominant in India. Whereas mostly luxurious brads like Lexus, BMW, Audi Volvo are the ones with most expensive cars.

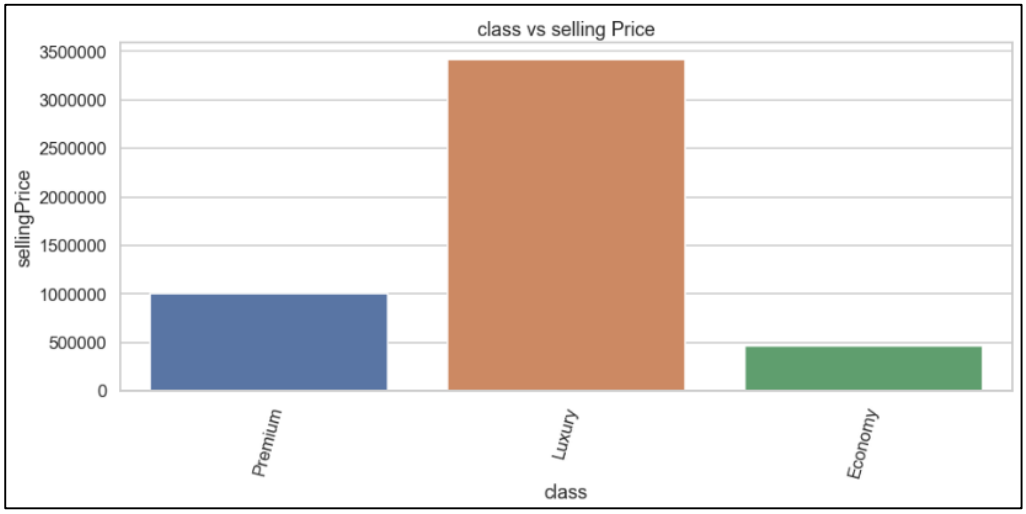
Looking at some of the specific cars that are known to belong to luxury, Premium and Economy categories I plotted their car price trend over the years and found observable differences.

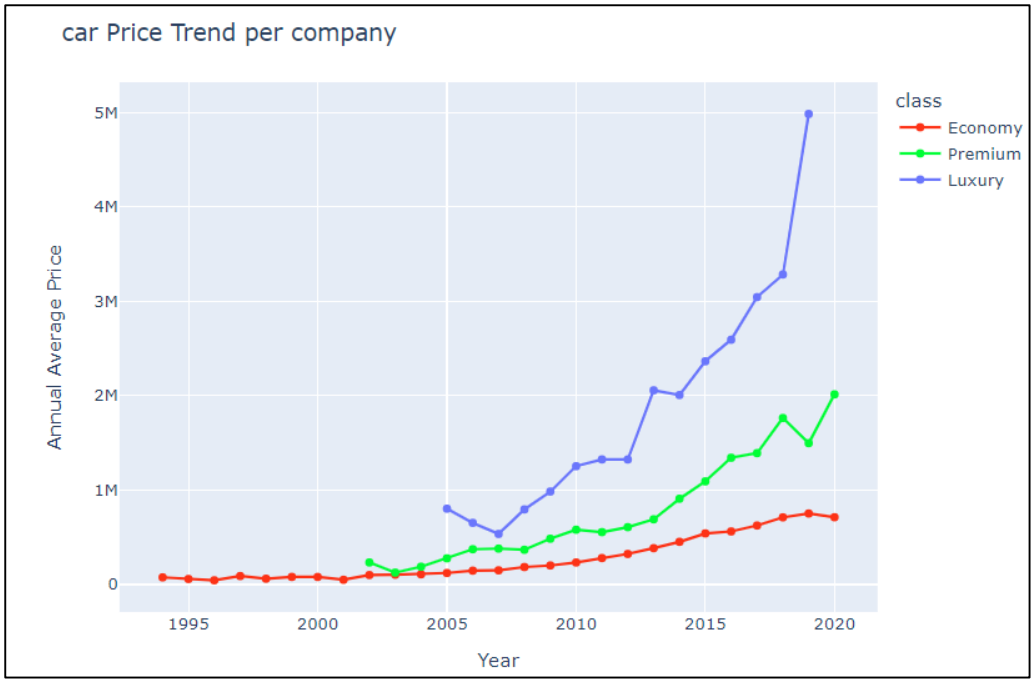


Looking into the derived features, country, continent and class

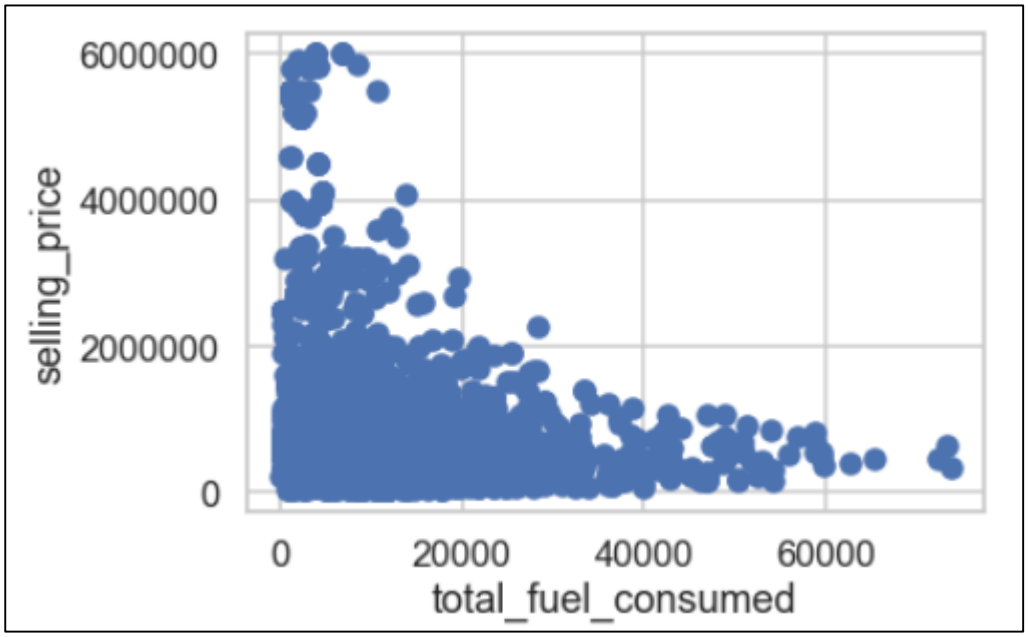

country vs selling Price
sellingPrice
3000000
2500000
2000000
1500000
1000000
500000
0
United States
United Kingdom
Sweden
South Korea
Japan
Italy
India
Germany
France
Czech Republic
country


continent vs selling Price
sellingPrice
1600000
1400000
1200000
1000000
800000
600000
400000
200000
0
North America
Europe
Asia
continent


class vs selling Price
sellingPrice
3500000
3000000
2500000
2000000
1500000
1000000
500000
0
Premium
Luxury
Economy
class



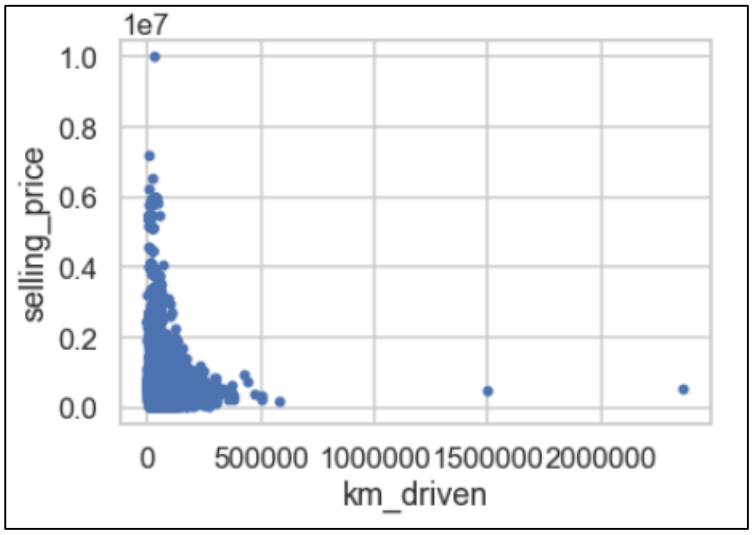
Fuel consumed by car: Using mileage and km driven I figured out total fuel consumed by a car



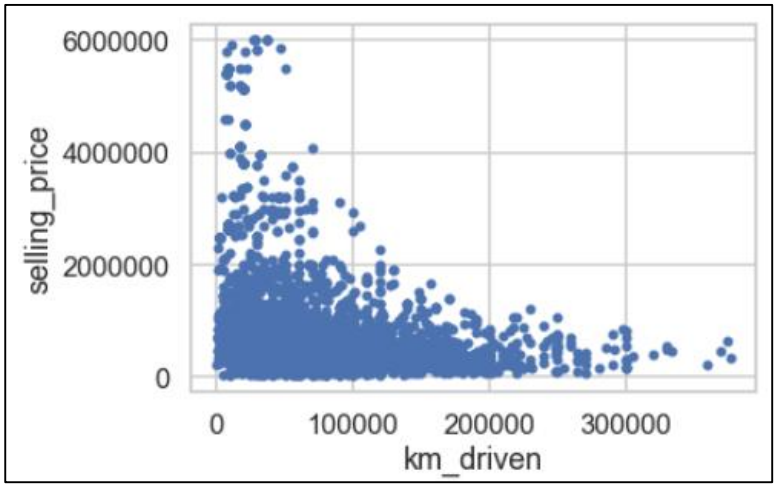
Removing Outliers:

There were a few outliers in the dataset mainly in Selling Price and KM_driven

Before



After



Dropping Extra columns: After feature engineering and creating some derived columns I dropped some of the extra columns. These columns that were dropped were year, name, km_driven, Years_used, fuel, owner, mileage, mileage_num, engine, engine_num, max_power, torque, seats, company_name, fuel_consumption.

One hot encoding Categorical Columns: There were a few categorical columns that I created during our Feature engineering and deriving new columns. These features had to be one hot encoded some of these columns are seller_type, transmission, year_category, fuel_new, owner_new, seats_Category, Engine_Category, mileage_Category, country, class, continent

Model Training and Fit –

Linear Regression: Before starting to fit with deep neural network, I wanted to first see how a simple linear regression performs with the dataset to be able to compare the results out of the DNN, so that I can use it as a benchmark.

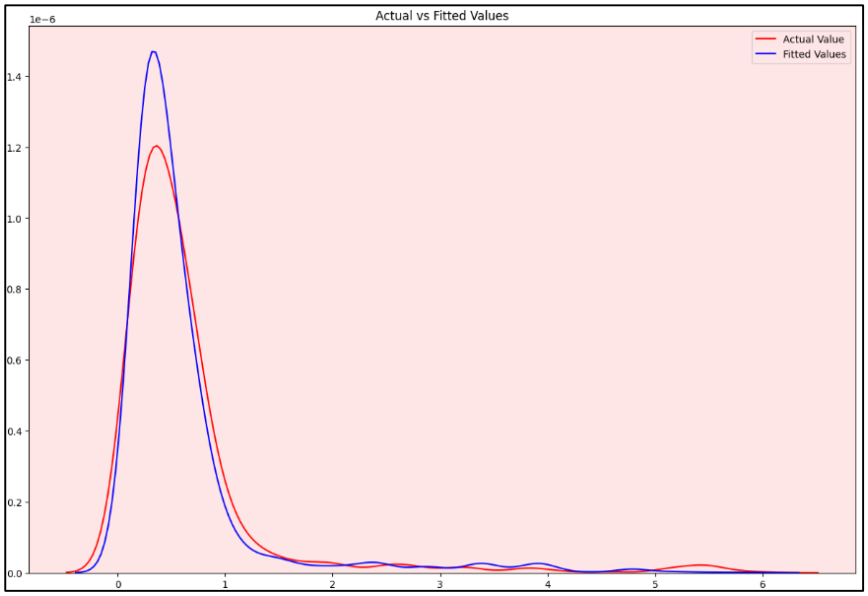
Using *Simple Linear Regression,* I was able to get an accuracy of ***76%***

At that point I tried with transforming the Y variable that is selling price using log transformation and I was able to achieve an accuracy of ***81%***, I will be using this as my bench mark.

Some of the important variables that came up as part of the Linear regression were –

- Engine Category variables
- Class Economy
- Fuel Type
- Continent Asia
- Country India
- Seller Type variables etc.

Fitting the predicted values against the actual values looks something like this with the log transformed variables



Overall it looks like that the fit has been good and it is not always possible to get very high prediction accuracies with predicting price or usage models, so I can tell that Linear regression model is working really nicely with our dataset.

Deep Neural Network Models: I first tried fitting a single layer with 160 nodes and looked for accuracies and I was not being able to come anywhere near to my bench mark.

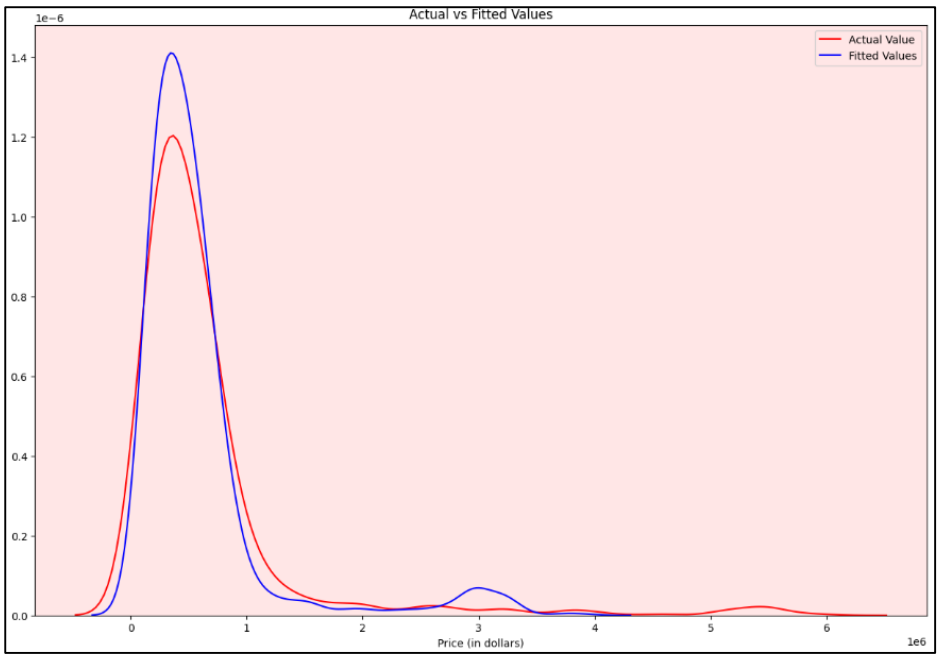
DNN Model 1

Finally, after playing around with the hidden layers and the nodes in each layer I was able to get an accuracy of ***63%*** with 3 hidden layers,

- Layer 1 – 160 nodes, drop out of 0.2, activation Relu
- Layer 2 – 480 nodes, drop out of 0.2, activation Relu

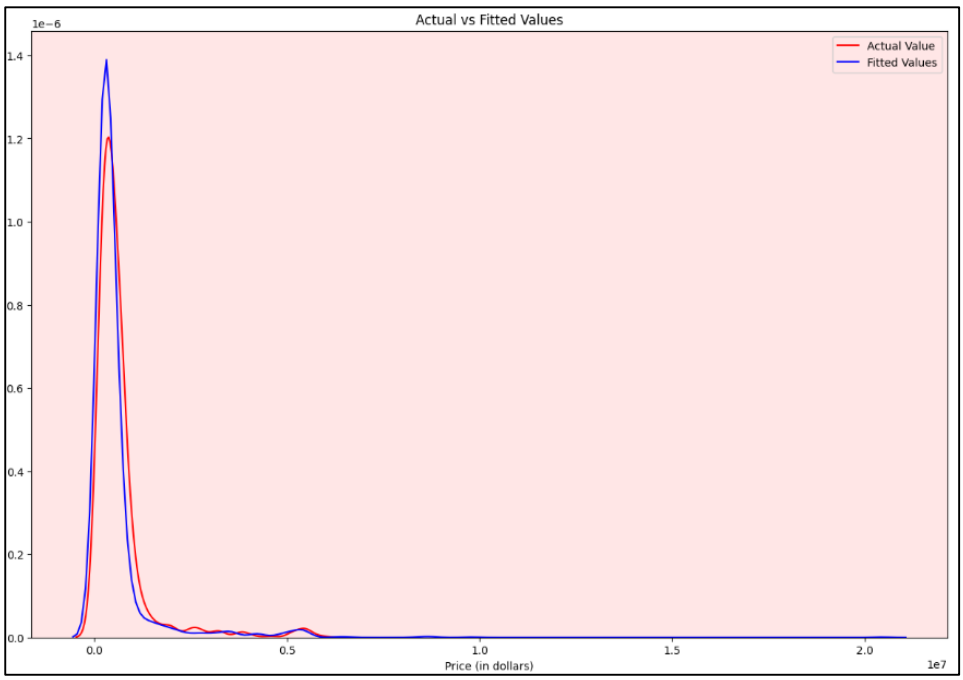
- Layer 3 – 256 nodes, drop out of 0.2, activation Relu
- Output Layer – 1 output node with activation Linear
- Learning rate used was 0.01
- Epochs = 50
- Batch size 64

For this I had used the actual Y variable without any transformation



Log transformed Y variable

Next I fitted the same model with the log transformed Y variable and keeping all the hyper parameters the same I was able to get an accuracy of ***71%***, still quite far away from the benchmark.



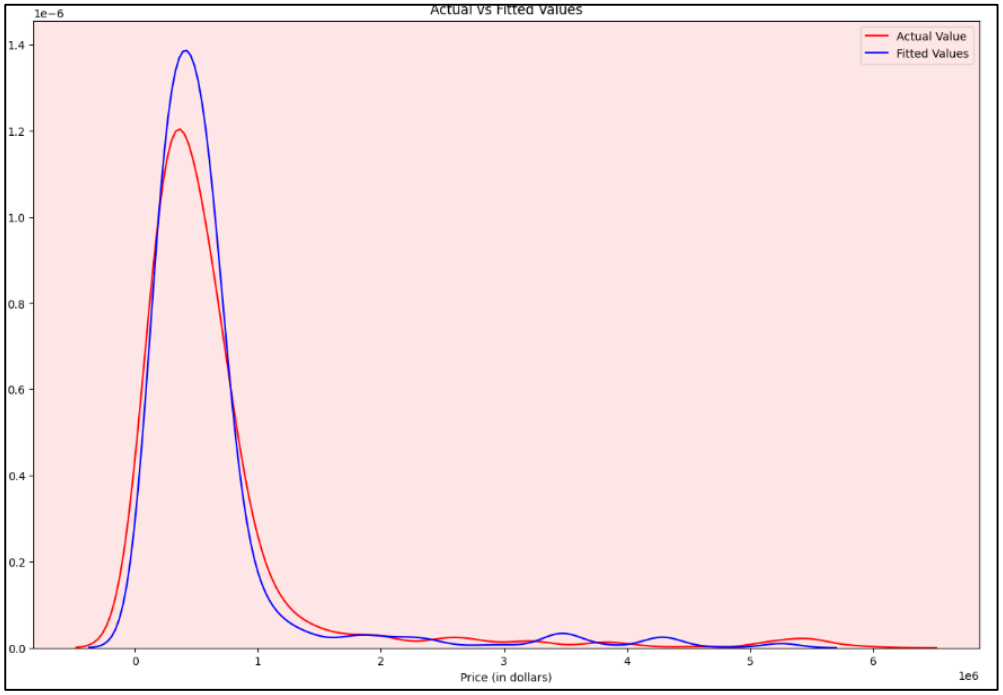
Since working with the log transformed Y variable is working better for us both in Linear regression and also in DNN, therefore rest of the models I used the log transformed Y variable only

DNN Model 2

Next I tried playing around with the layers and the number of nodes and was able to get an accuracy of ***86%***, with this new accuracy I was able to do better than my bench mark of Linear regression which was 81%

The hyper parameters utilized for this model

- Layer 1 – 80 nodes, drop out of 0.2, activation Relu
- Layer 2 – 128 nodes, drop out of 0.2, activation Relu
- Layer 3 – 90 nodes, drop out of 0.2, activation Relu
- Output Layer – 1 output node with activation Linear
- Learning rate used was 0.01
- Epochs = 50
- Batch size 64

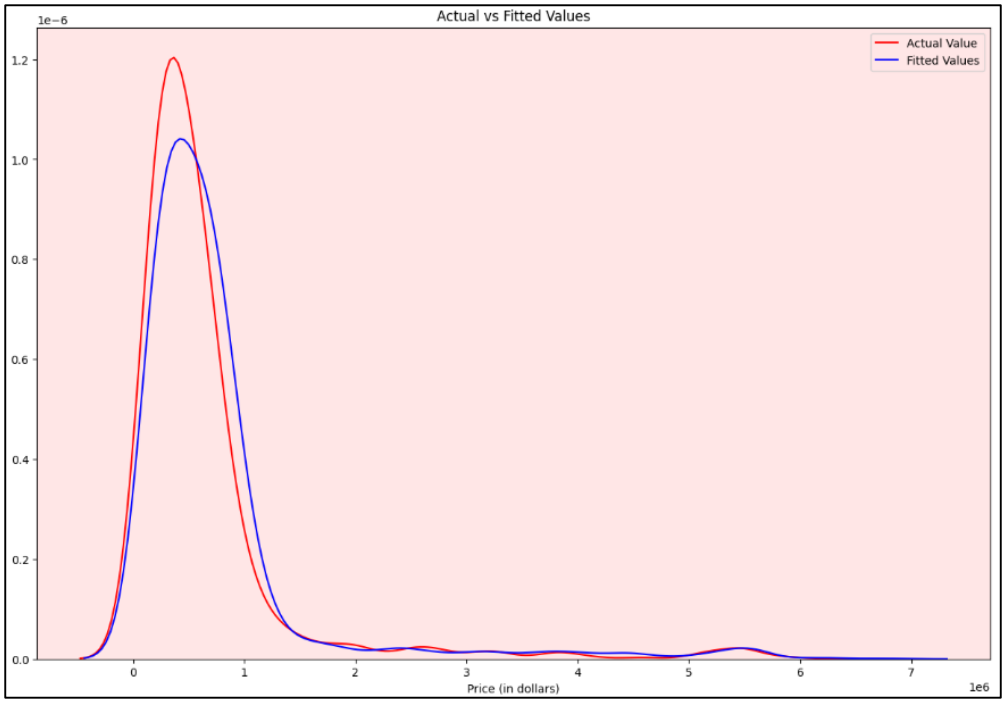


Next I wanted to check if there is any chance of increasing the accuracy even further using regularization techniques.

Using Regularization:

Keeping rest of the hyper parameters the same I applied L1 regularization with alpha = 0.001 and was able to get an accuracy of ***87%,***

Note: Along with the models that are being mentioned in this document I had tried multiple other variations but was not able to get an accuracy higher than 87%



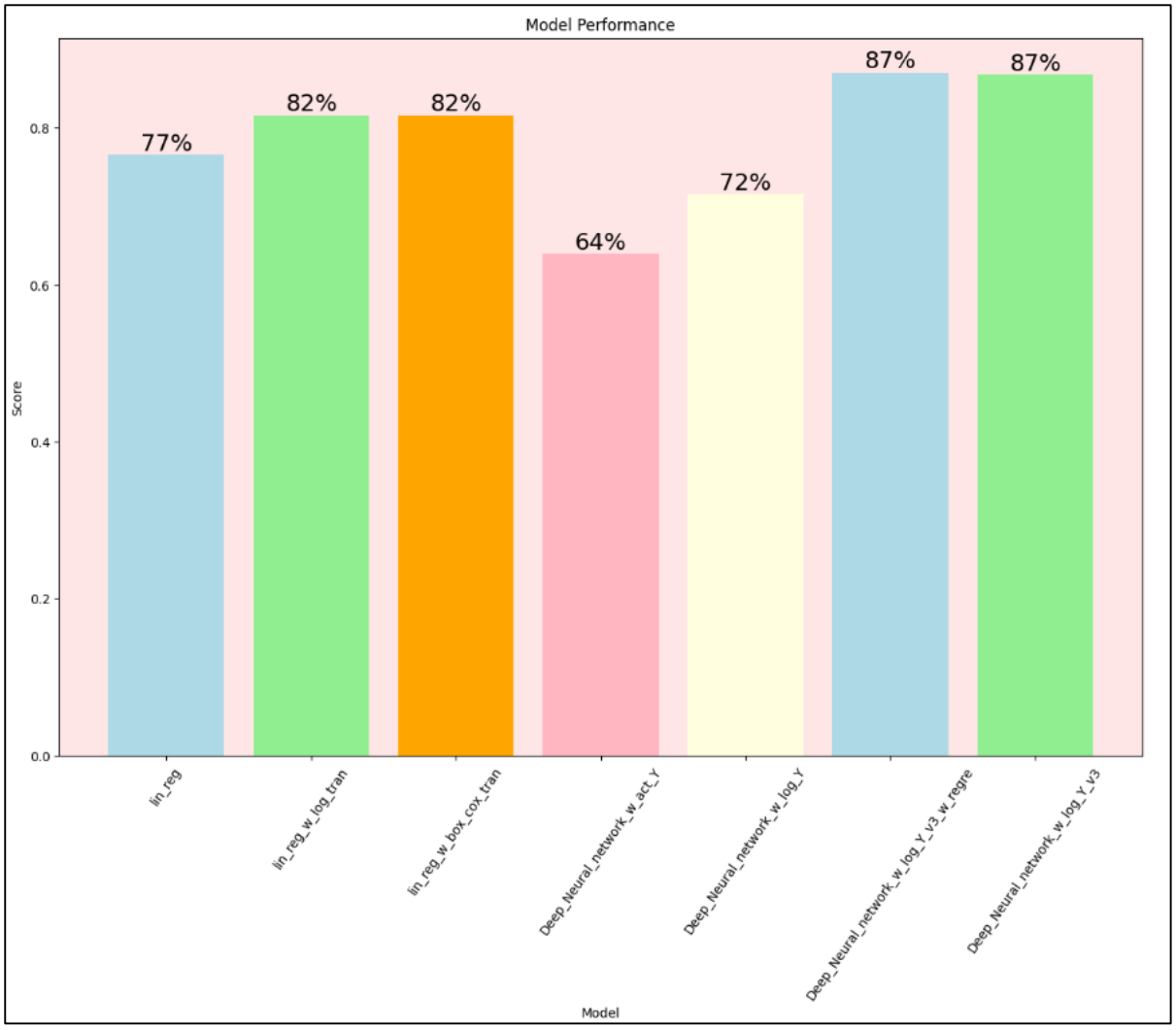
Summary of Models

with basic linear regression model we get an accuracy of 76%, but on log transforming the Y variable we get a better accuracy of 81%

Using Deep neural network with Actual Y gave us a low score of 63 % but on tuning the model hyper parameters and using the log transformation of the Y variable we were able to increase the accuracy to 71%

Finally, with fine tuning the hyper parameters and the regularizing parameters I was able to achieve an accuracy of 87%

For all my models I have utilized ADAM optimizer



Next Steps

Overall I was able to build a Deep neural network model with my understanding and achieve an accuracy of 87%, but I feel since most of the data contained of categorical variables with only km driven and max power being the only two continuous variables it was hard to derive much more information that could be learned.

I would like to collect some more data and retrain the models. Also simple linear regression give accuracy of 81% tells me that the data is quite linear and we might be able to do better using some more linear models. I will be trying to see if some other variations of the Neural network is able to learn about the linearity in the data.

I would also keep on experimenting with more variations of the hidden layers to figure out if there can be a better model that gives us accuracy in the range of 90%.