FLIP ROBO

CAR PRICE PREDICTION

Submitted by:

P. MANIVANNAN

INTRODUCTION

Problem Statement:

With the covid 19 impact in the market, we have seen lot of changes in the car market. Now some cars are in demand hence making them costly and some are not in demand hence cheaper. One of our clients works with small traders, who sell used cars. With the change in market due to covid 19 impact, our client is facing problems with their previous car price valuation machine learning models. So, they are looking for new machine learning models from new data. We have to make car price valuation model.

- Conceptual Background of the Domain Problem

Houses Due to Covid-19 Impact the Old machine learning models are not performing well . so we need new model based on recent data to analyse the Current Sentiment of 2nd Cars Market.

- Review of Literature

The Research is done on Used Car Sales Market and how each variable play a role in Price of the Car with Data Science as tool the research is done. First Brief Exploratory Data Analysis is done which includes Handling Null values, Skewness Removal, Outlier Treatment, Visualizing the Data to Understand Correlation with Car price and understanding Relation between various input features. Then Data is Fed to Machine Learning Algorithms. after Evaluating and Testing Different Algorithms Best Algorithm is Selected. Based on Evaluation Metrics such as Training Score, R2 Score, Cross Validation Score etc.

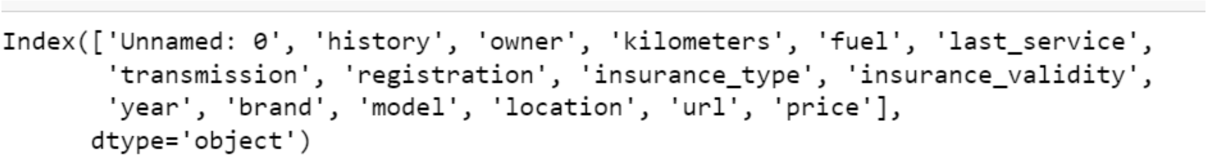
- Motivation for the Problem Undertaken

The Car price is always a mystery. India is huge market for used cars. So this motivated me to understand what parameters affect the price of the used Cars.

Analytical Problem Framing

- Data Sources and their formats

The data contains 5610 rows and 16 columns. The columns are



```
Index(['Unnamed: 0', 'history', 'owner', 'kilometers', 'fuel', 'last_service',
       'transmission', 'registration', 'insurance_type', 'insurance_validity',
       'year', 'brand', 'model', 'location', 'url', 'price'],
      dtype='object')
```

The used cars data is collected from cars24 website using web scrapping. data is collected from different cities in India and combined.

The data is in xlsx format.

- Data Pre-processing Done

1) The data has null values which is removed using dropna method of pandas. As the null values are in different brands models . if we use mean method to fill them it will bias the model so null values are dropped. which is only 1.8% data removed as null values .
2) We had categorical data which needs to be encoded. for this approach I used pd.get_dummies method of pandas library as categorical data are not ordinal in nature.
3) Then Skewness Is checked for the data and Skewness is removed using power_transform function.
4) Then outliers is checked using Z Score method. And removed.
5) The data are then standardized using Standard Scaler.
6) As the columns count increased after encoding with dummies method. I decided to used Principal Component Analysis to reduce the No of columns. To decide the No of columns I used pca.explained_variance_ratio_ method to find how much each feature (maximum variance) contribute to dataset.

- Data Inputs- Logic- Output Relationships

Using Correlation function corr(). The correlation of features is analysed and how it affects our target variable Car Price. I found that Kilometers Feature has negative correlation with Target Variable . then the owner feature is also negatively correlated with target variable as the no of owner increases the price of car decreases. And for fuel type diesel car price is higher compared to other variants. And transmission feature the auto transmission price is higher compared to manual. The year of car plays a important role in price . as the car gets older the price decreases. Compared to cars purchased at recent years.

- State the set of assumptions (if any) related to the problem under consideration

1) For Handling the Null values . Null values are dropped considering they will not affect the prediction.
2) For handling the categorical data dummies method is used assuming the categorical data do not have ordinal relationship with each other. as more insight is need to conclude which category is ordinal or not. its safe to use dummies method.
3) Skewness is removed assuming the prediction is not affected by skewness removal.
4) Outliers are removed, assuming that those data are really outliers and do not contribute to dataset
5) Principal component analysis technique is used to reduce the column count. But the variance of features are retained by using pca.explained_variance_ratio_ method to find the variance exhibited by different features.

- Hardware and Software Requirements and Tools Used

The Hardware requirements are 16 Gb RAM,500 GB SSD, at least 6 Cores processor

The software requirements are Windows, Anaconda Framework and libraries used are Pandas, NumPy, Sklearn, SciPy, Matplotlib, Seaborn

We use Pandas for reading the Dataset, Creating Dataframe, much more to handle data in dataset

Certain functions of NumPy are used like log , cbrt ,sqrt skewness transformation . absolute (abs) function used along with Zscore.

SKlearn Library is used for Machine Learning Algorithms like SVC, Random Forest, Linear Regression etc and Metrics for analysing them.

Scipy is scientific python library used for Zscore method etc.

Matplotlib and Seaborn are used for visualizations plotting graphs charts, etc

Finally joblib module is used for saving our model for future use.

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

  Since the Target Variable is Continuous in nature I should Use Regression Approach to Solve this Problem and build a Regression Model

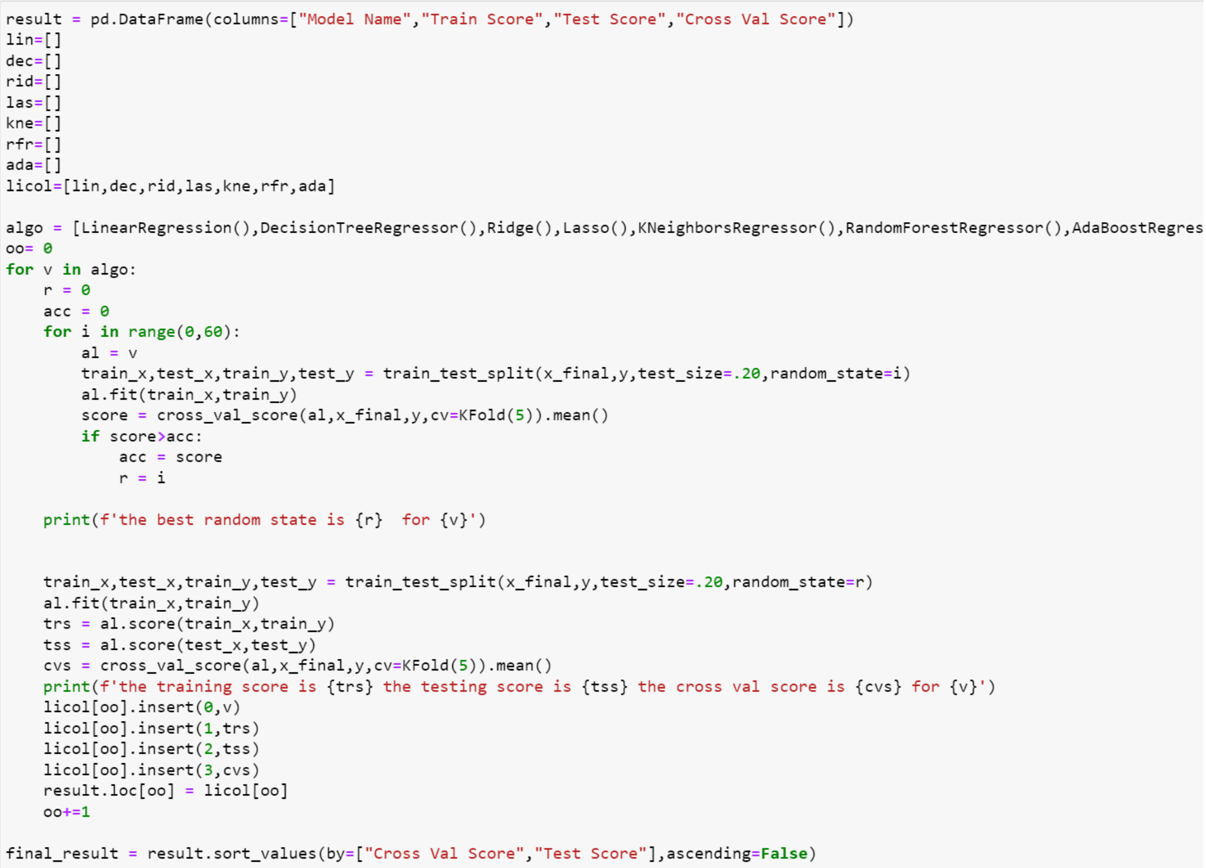
- Testing of Identified Approaches (Algorithms)
  1) Linear Regression

2) Ridge Regression
3) Lasso Regression
4) Decision Tree Regressor
5) Random Forest Regressor
6) Ada Boost Regressor
7) KNeighbours Regressor

- Run and Evaluate selected models

  I have custom written a Code Block for Running all Algorithms and Storing them in DataFrame and Ranking them based on their Accuracy Score. The best part is the code finds the best random state for each algorithm and prints them

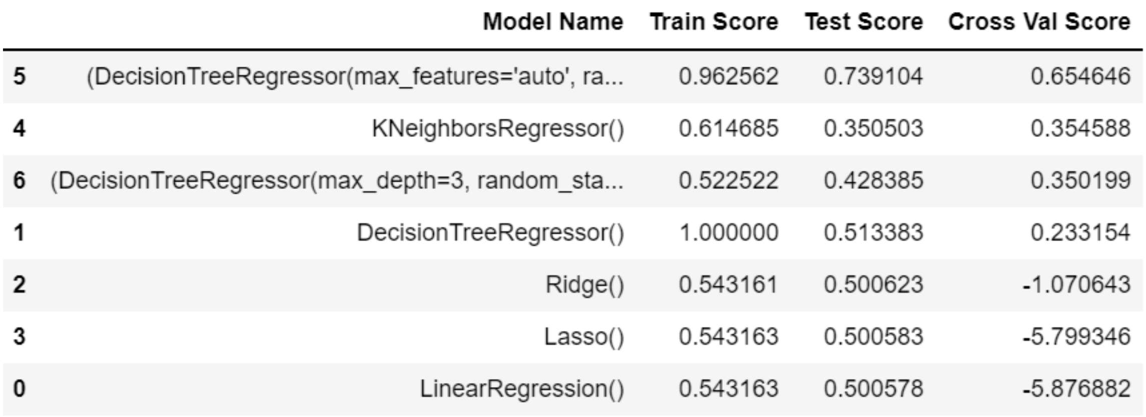
  Code:



```
result = pd.DataFrame(columns=["Model Name","Train Score","Test Score","Cross Val Score"])
lin=[]
dec=[]
rid=[]
las=[]
kne=[]
rfr=[]
ada=[]
licol=[lin,dec,rid,las,kne,rfr,ada]

algo = [LinearRegression(),DecisionTreeRegressor(),Ridge(),Lasso(),KNeighborsRegressor(),RandomForestRegressor(),AdaBoostRegres
oo= 0
for v in algo:
    r = 0
    acc = 0
    for i in range(0,60):
        al = v
        train_x,test_x,train_y,test_y = train_test_split(x_final,y,test_size=.20,random_state=i)
        al.fit(train_x,train_y)
        score = cross_val_score(al,x_final,y,cv=KFold(5)).mean()
        if score>acc:
            acc = score
            r = i

    print(f'the best random state is {r}  for {v}')


    train_x,test_x,train_y,test_y = train_test_split(x_final,y,test_size=.20,random_state=r)
    al.fit(train_x,train_y)
    trs = al.score(train_x,train_y)
    tss = al.score(test_x,test_y)
    cvs = cross_val_score(al,x_final,y,cv=KFold(5)).mean()
    print(f'the training score is {trs} the testing score is {tss} the cross val score is {cvs} for {v}')
    licol[oo].insert(0,v)
    licol[oo].insert(1,trs)
    licol[oo].insert(2,tss)
    licol[oo].insert(3,cvs)
    result.loc[oo] = licol[oo]
    oo+=1

final_result = result.sort_values(by=["Cross Val Score","Test Score"],ascending=False)
```

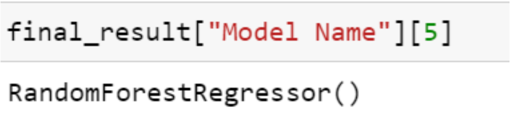
If you see the code I have created a list of algorithm which I need to run. And here is the output:

Output1:



	Model Name	Train Score	Test Score	Cross Val Score
5	(DecisionTreeRegressor(max_features='auto', ra...	0.962562	0.739104	0.654646
4	KNeighborsRegressor()	0.614685	0.350503	0.354588
6	(DecisionTreeRegressor(max_depth=3, random_sta...	0.522522	0.428385	0.350199
1	DecisionTreeRegressor()	1.000000	0.513383	0.233154
2	Ridge()	0.543161	0.500623	-1.070643
3	Lasso()	0.543163	0.500583	-5.799346
0	LinearRegression()	0.543163	0.500578	-5.876882

Output2:



```
final_result["Model Name"][5]
```

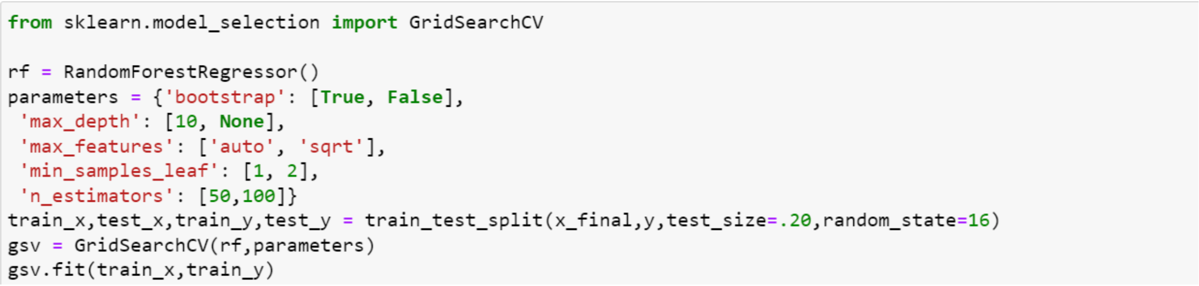
```
RandomForestRegressor()
```

Here you can see the Model Name, Training Score, Testing Score.

Based on this best model is selected and further proceeded for hyperparameter tuning. Here Ridge Regression performed best compared to others.

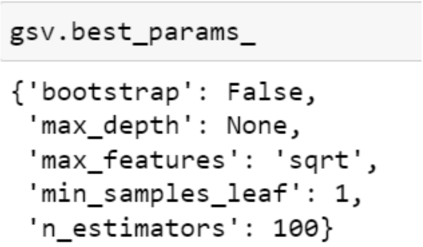
Hyperparameter Tuning

Code:



```
from sklearn.model_selection import GridSearchCV

rf = RandomForestRegressor()
parameters = {'bootstrap': [True, False],
 'max_depth': [10, None],
 'max_features': ['auto', 'sqrt'],
 'min_samples_leaf': [1, 2],
 'n_estimators': [50,100]}
train_x,test_x,train_y,test_y = train_test_split(x_final,y,test_size=.20,random_state=16)
gsv = GridSearchCV(rf,parameters)
gsv.fit(train_x,train_y)
```

Output:

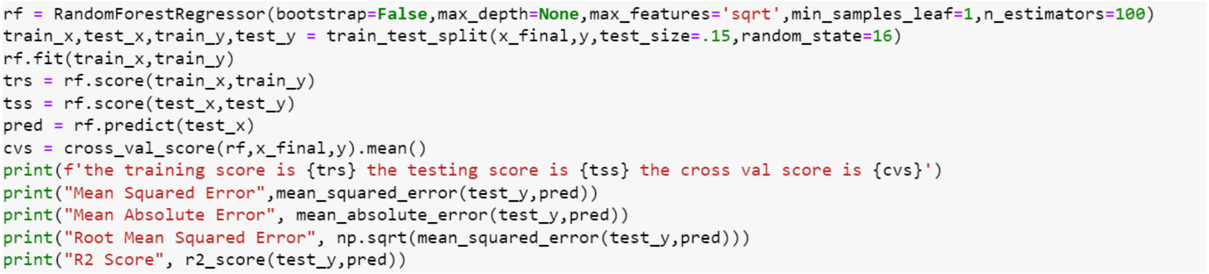


```
gsv.best_params_
```

```
{'bootstrap': False,
 'max_depth': None,
 'max_features': 'sqrt',
 'min_samples_leaf': 1,
 'n_estimators': 100}
```

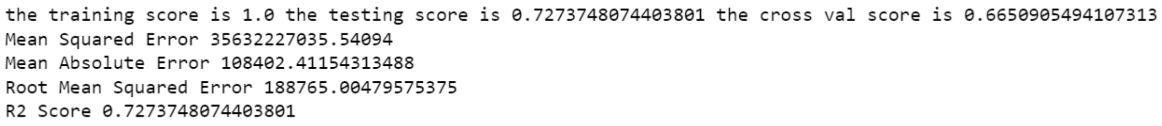
The Best Parameters are listed now we need to run the model with the best parameters

Code:



```
rf = RandomForestRegressor(bootstrap=False,max_depth=None,max_features='sqrt',min_samples_leaf=1,n_estimators=100)
train_x,test_x,train_y,test_y = train_test_split(x_final,y,test_size=.15,random_state=16)
rf.fit(train_x,train_y)
trs = rf.score(train_x,train_y)
tss = rf.score(test_x,test_y)
pred = rf.predict(test_x)
cvs = cross_val_score(rf,x_final,y).mean()
print(f'the training score is {trs} the testing score is {tss} the cross val score is {cvs}')
print("Mean Squared Error",mean_squared_error(test_y,pred))
print("Mean Absolute Error", mean_absolute_error(test_y,pred))
print("Root Mean Squared Error", np.sqrt(mean_squared_error(test_y,pred)))
print("R2 Score", r2_score(test_y,pred))
```

Output:



```
the training score is 1.0 the testing score is 0.7273748074403801 the cross val score is 0.6650905494107313
Mean Squared Error 35632227035.54094
Mean Absolute Error 108402.41154313488
Root Mean Squared Error 188765.00479575375
R2 Score 0.7273748074403801
```

The Best Training Score for Random forest Regressor After Hyper parameter tuning for Training is 100% , Testing is 72% and Cross Validation Score is 66%

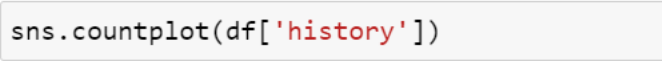
- Key Metrics for success in solving problem under consideration

Since it’s a Regression problem The key metrics used are Score method in Regression for training and testing, R2 Score, Mean Squared error, Mean Absolute Error, Root Mean Squared Error.

Visualizations

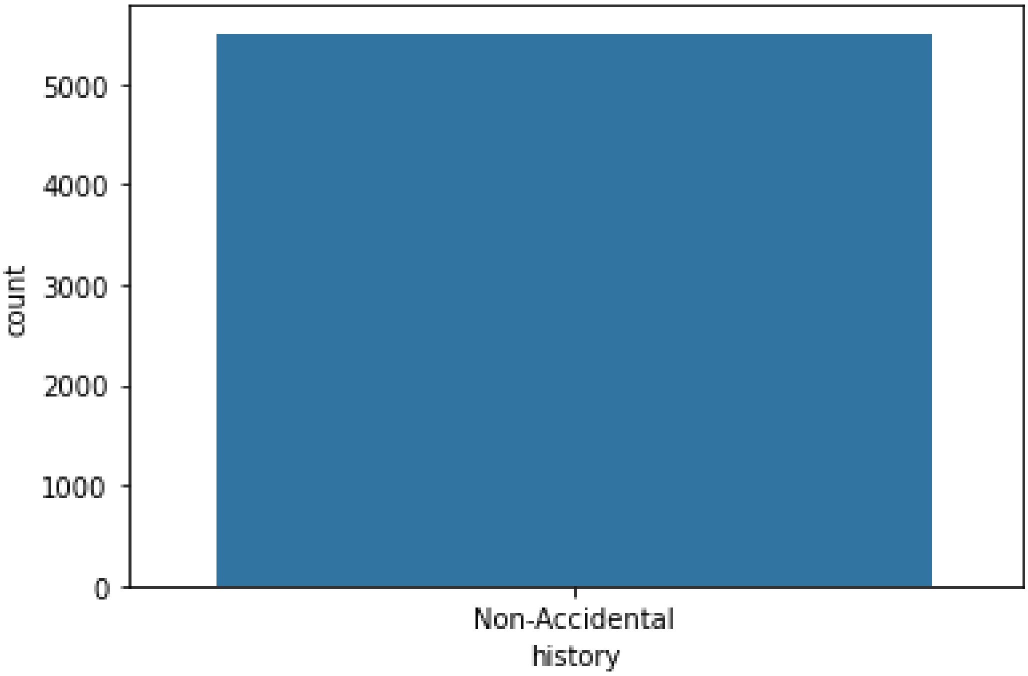
For visualizations the Matplotlib & Seaborn Library are used.

Code:



```
sns.countplot(df['history'])
```

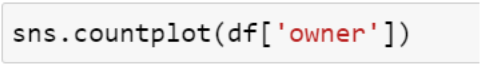
Output:



Observation:

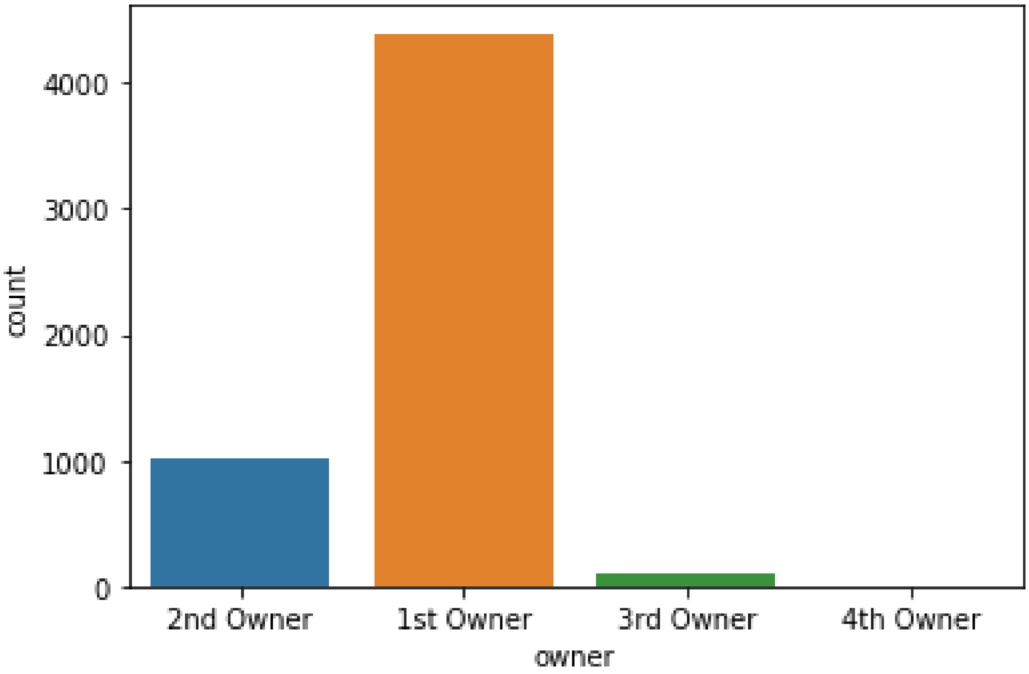
All the Car data is non- accidental

Code:



```
sns.countplot(df['owner'])
```

Output:



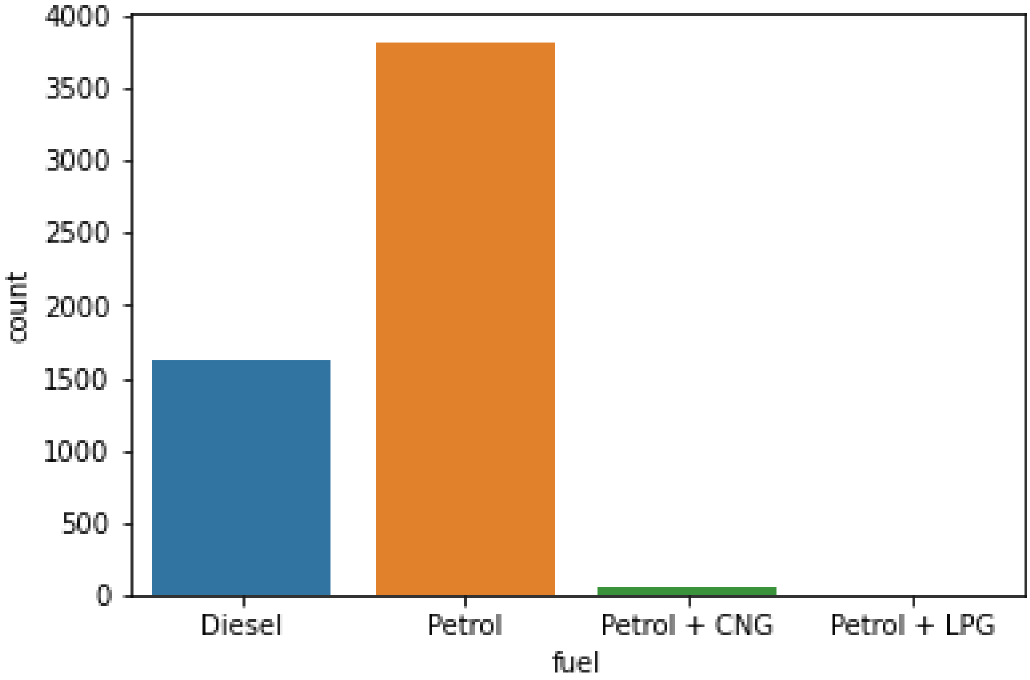
Observation:

The 1st Owner Car Data is higher compared to others

Code:

```
sns.countplot(df['fuel'])
```

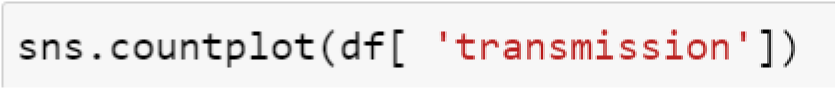
Output:



Observation:

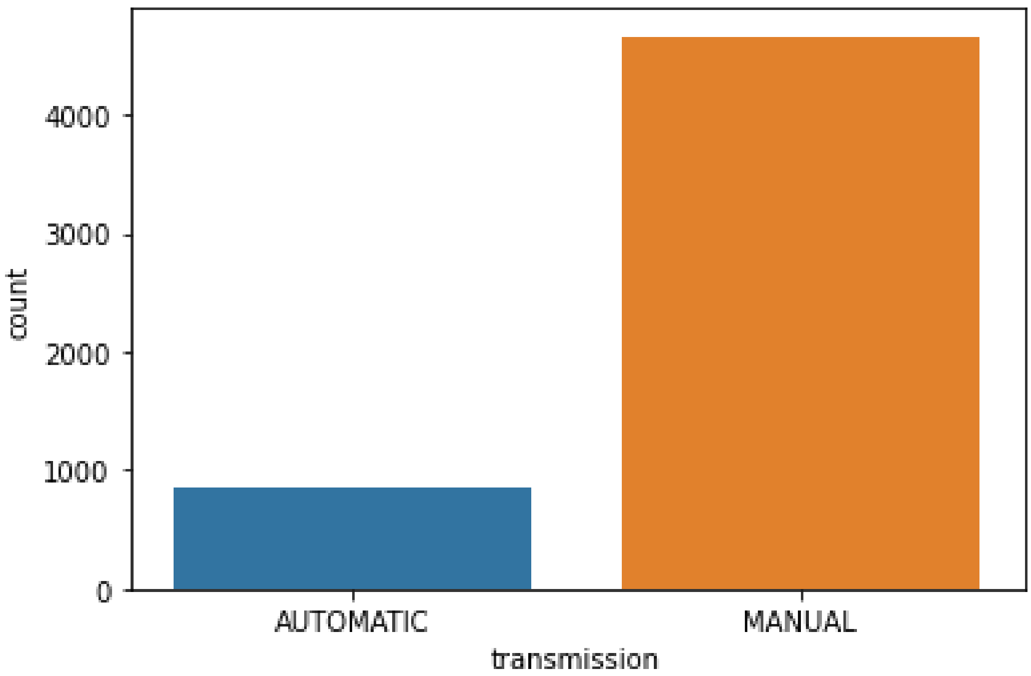
Petrol vehicles are higher in number compared to other fuel types

Code:



```
sns.countplot(df[ 'transmission'])
```

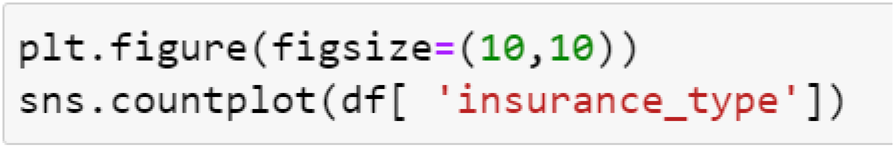
Output:



Observation:

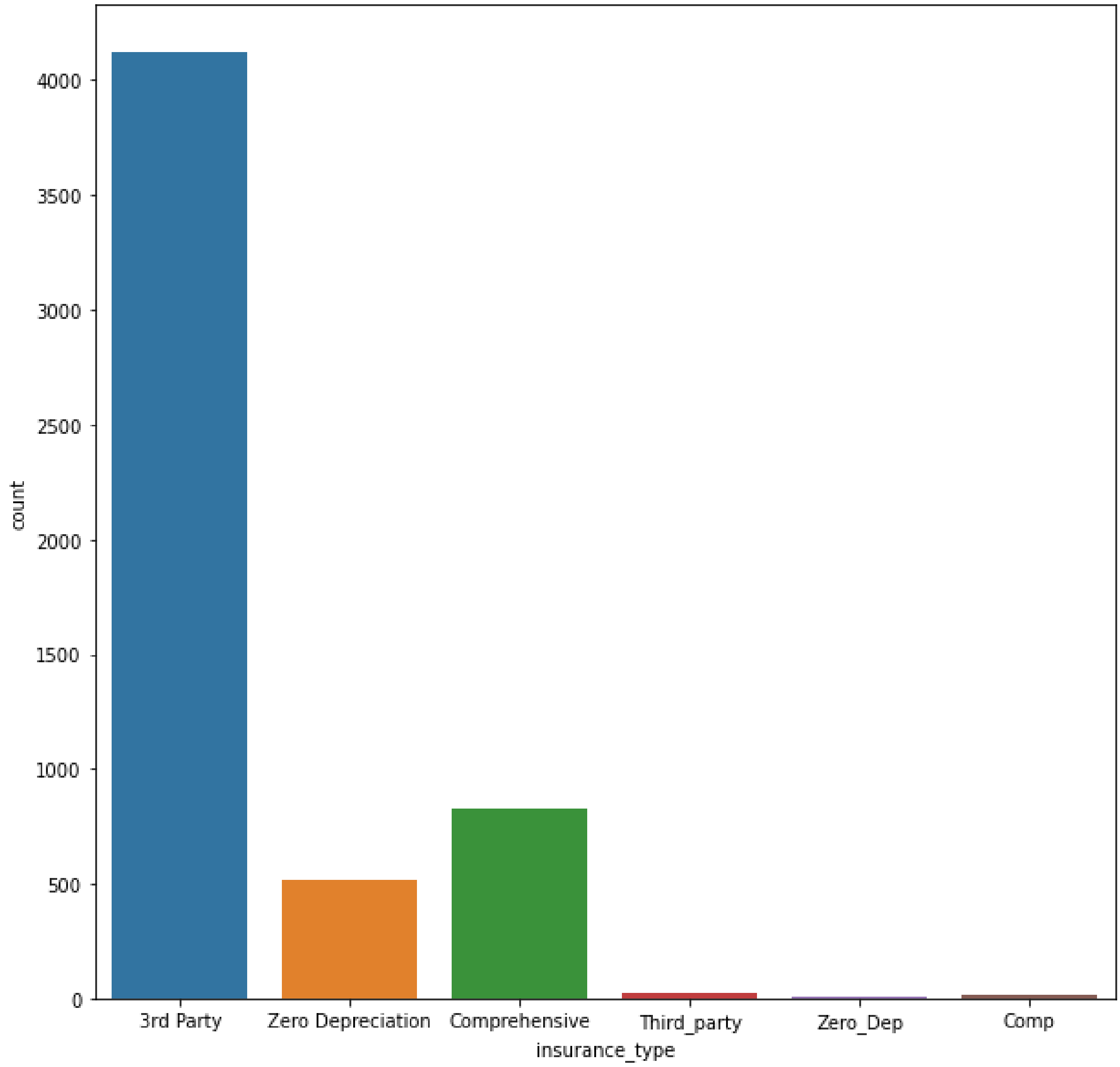
Manual Transmission is higher in cars compared to Automatic

Code:



```
plt.figure(figsize=(10,10))
sns.countplot(df[ 'insurance_type'])
```

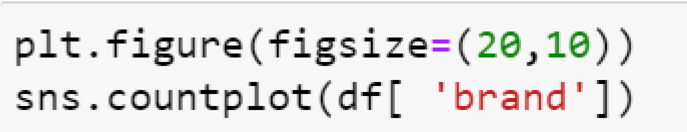
Outpu



Observation:

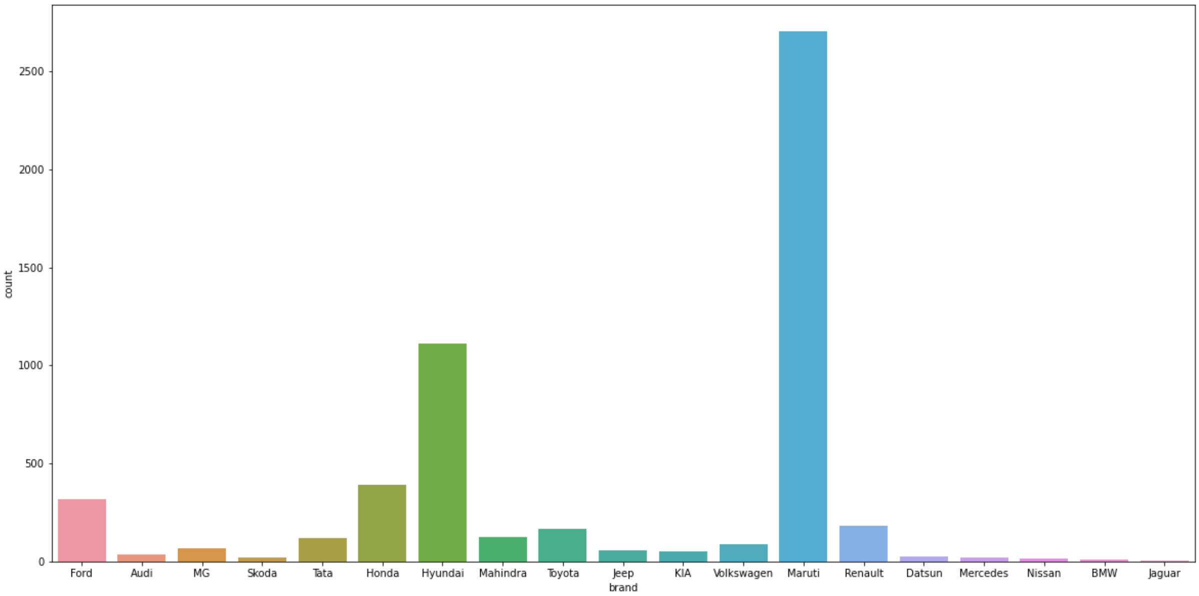
3rd Party Data insurance is common among most of the cars

Code:



```
plt.figure(figsize=(20,10))
sns.countplot(df[ 'brand'])
```

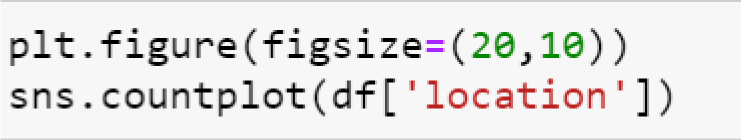
Output:



Observation:

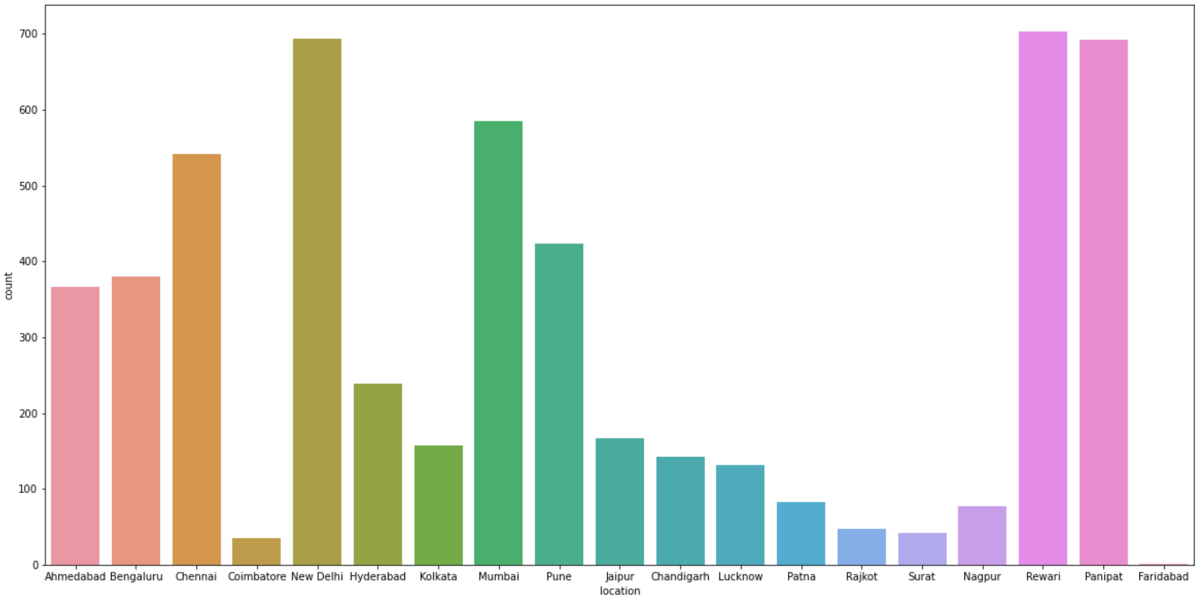
Maruthi Car is higher in 2nd Data car sales Data

Code:



```
plt.figure(figsize=(20,10))
sns.countplot(df['location'])
```

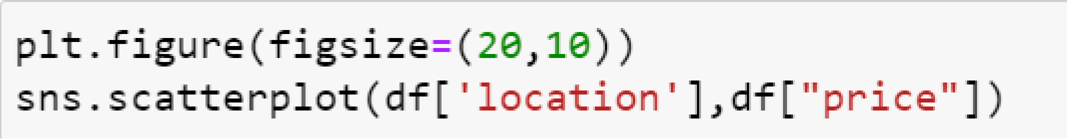
Output:



Observation:

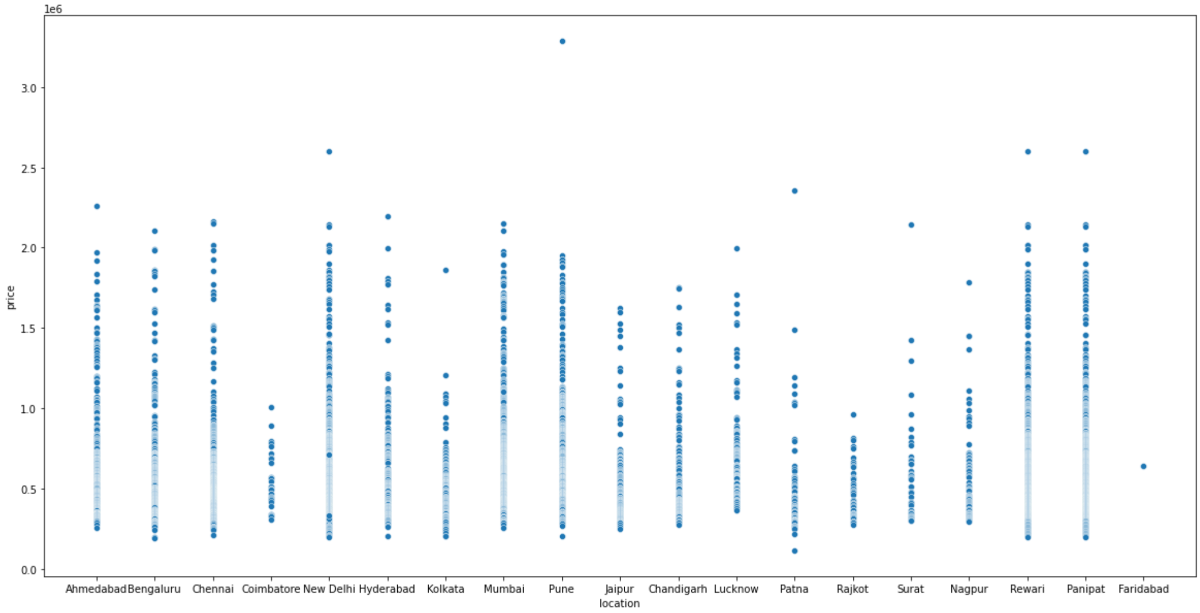
New Delhi has higher 2nd car listings

Code:



```
plt.figure(figsize=(20,10))
sns.scatterplot(df['location'],df["price"])
```

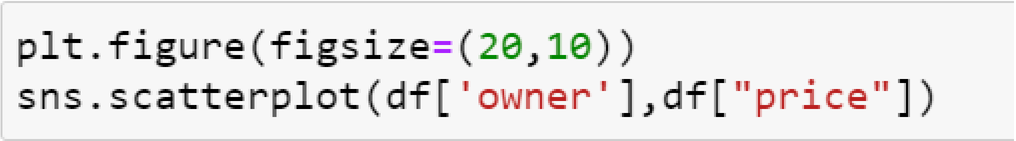
Output:



Observation:

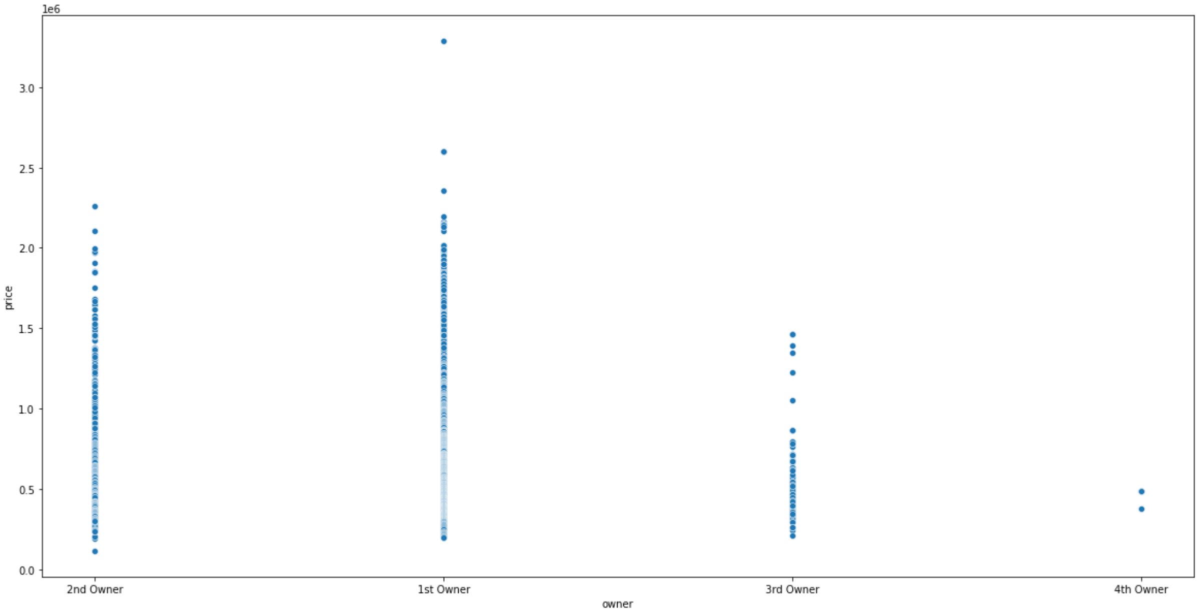
Coimbatore , Rajkot , Faridabad, Kolkata the price is lower compared to other cities

Code:



```
plt.figure(figsize=(20,10))
sns.scatterplot(df['owner'],df["price"])
```

Output:



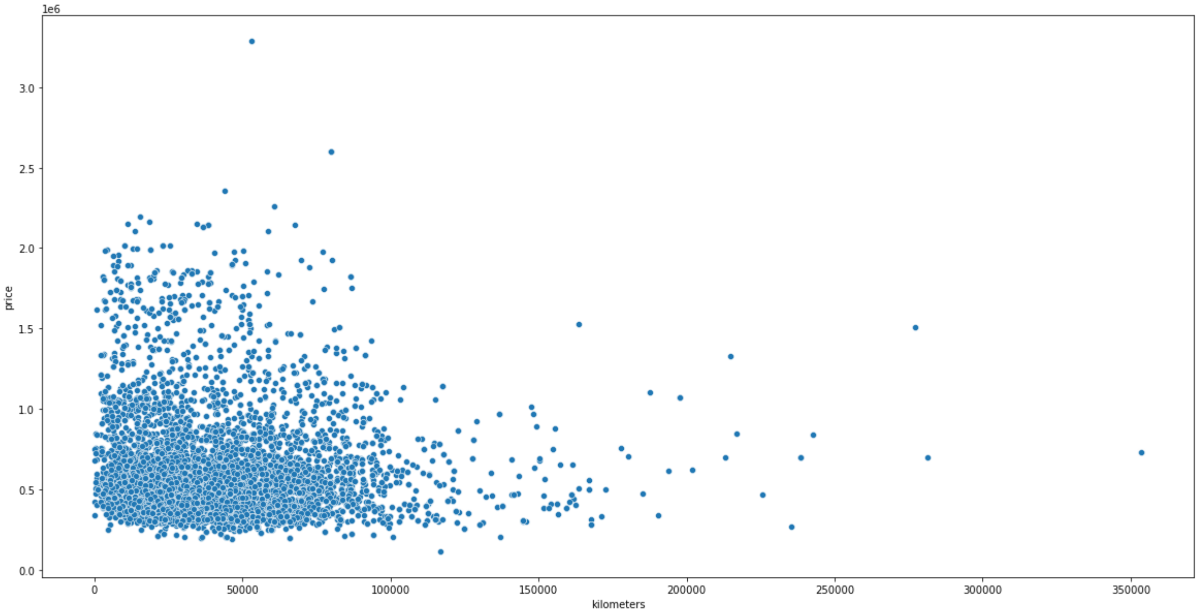
Observation:

The Car Price is Higher for the 1st owner cars. people prefer cars with 1st owner

Code:

```
plt.figure(figsize=(20,10))
sns.scatterplot(df['kilometers'],df["price"])
```

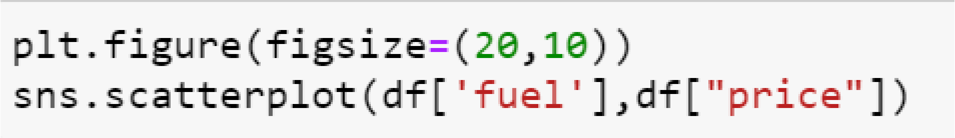
Output:



Observation:

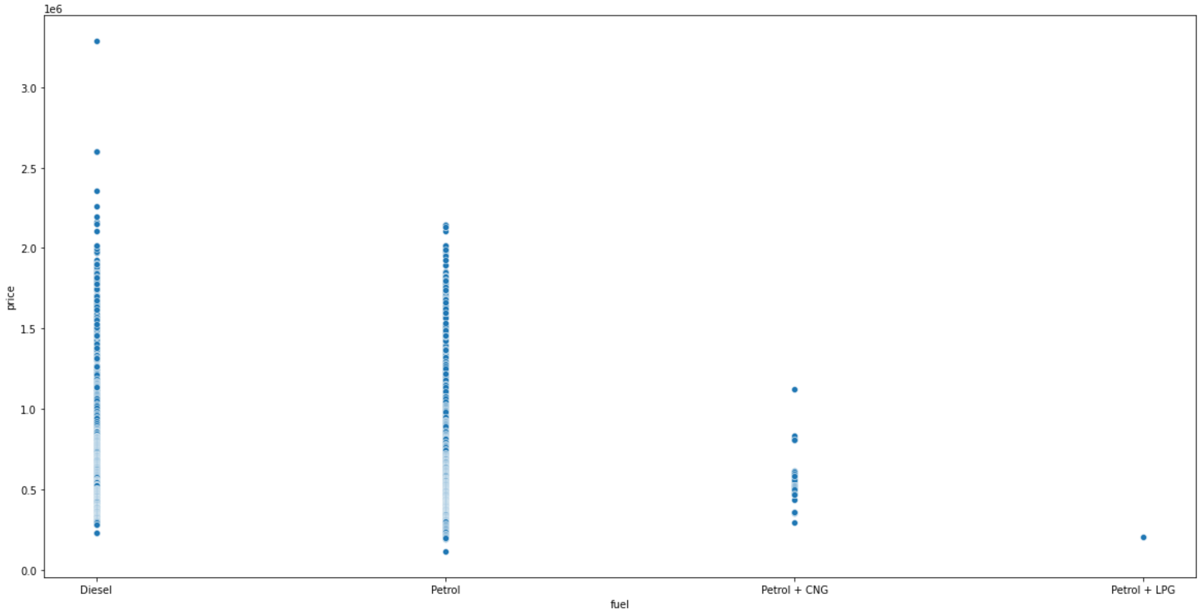
The Price Decrease with increase in Kilometres .

Code:



```
plt.figure(figsize=(20,10))
sns.scatterplot(df['fuel'],df["price"])
```

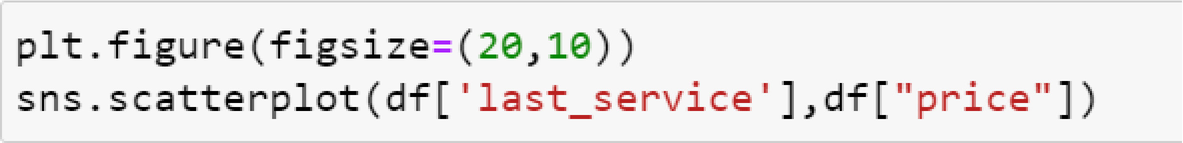
Output:



Observation:

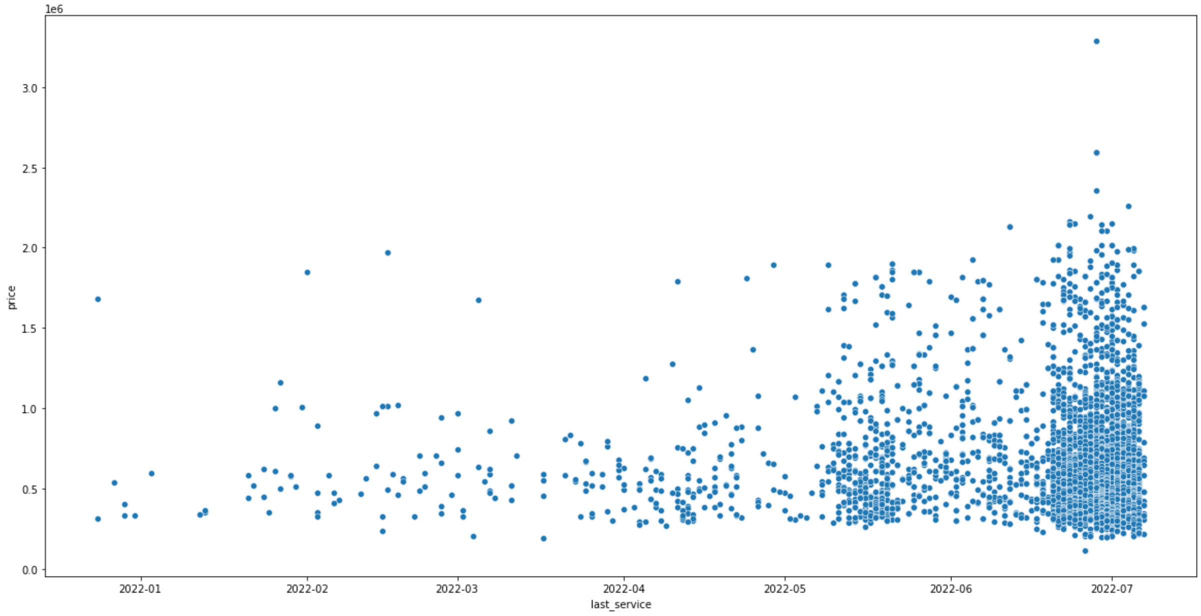
The Diesel cars price are higher compared to other fuel types next is petrol cars

Code:



```
plt.figure(figsize=(20,10))
sns.scatterplot(df['last_service'],df["price"])
```

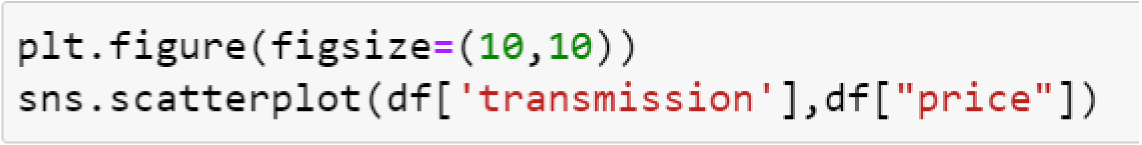
Output:



Observation:

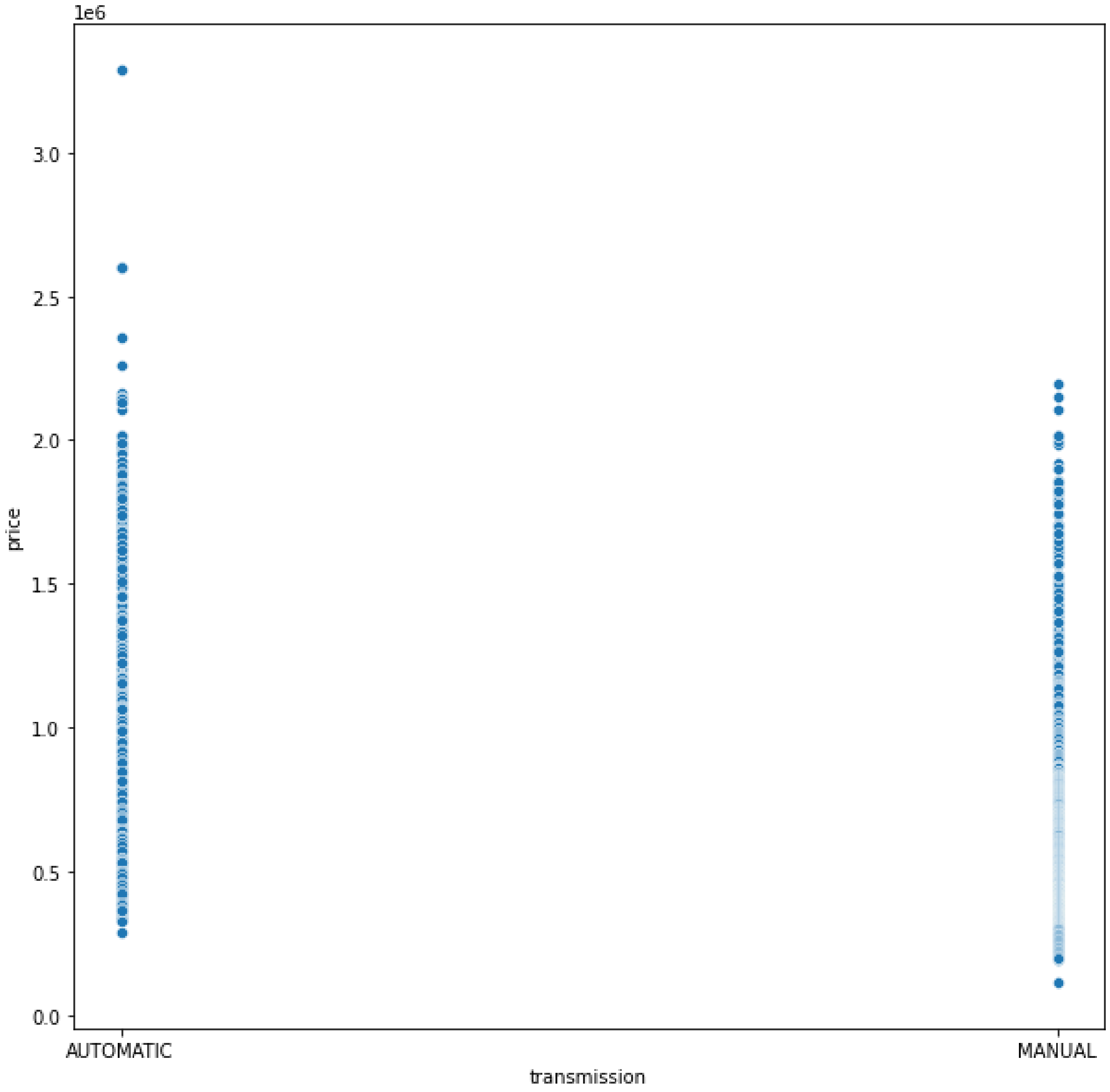
Cars Serviced long before have very low price compared to cars serviced recently

Code:



```
plt.figure(figsize=(10,10))
sns.scatterplot(df['transmission'],df["price"])
```

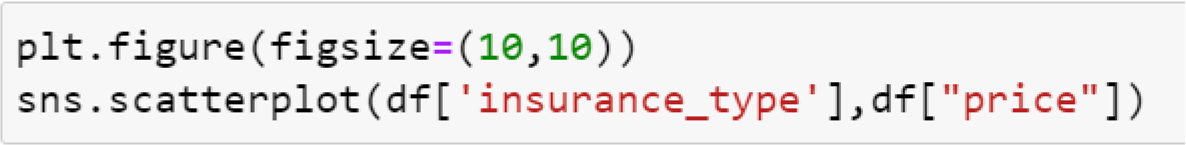
Output:



Observation:

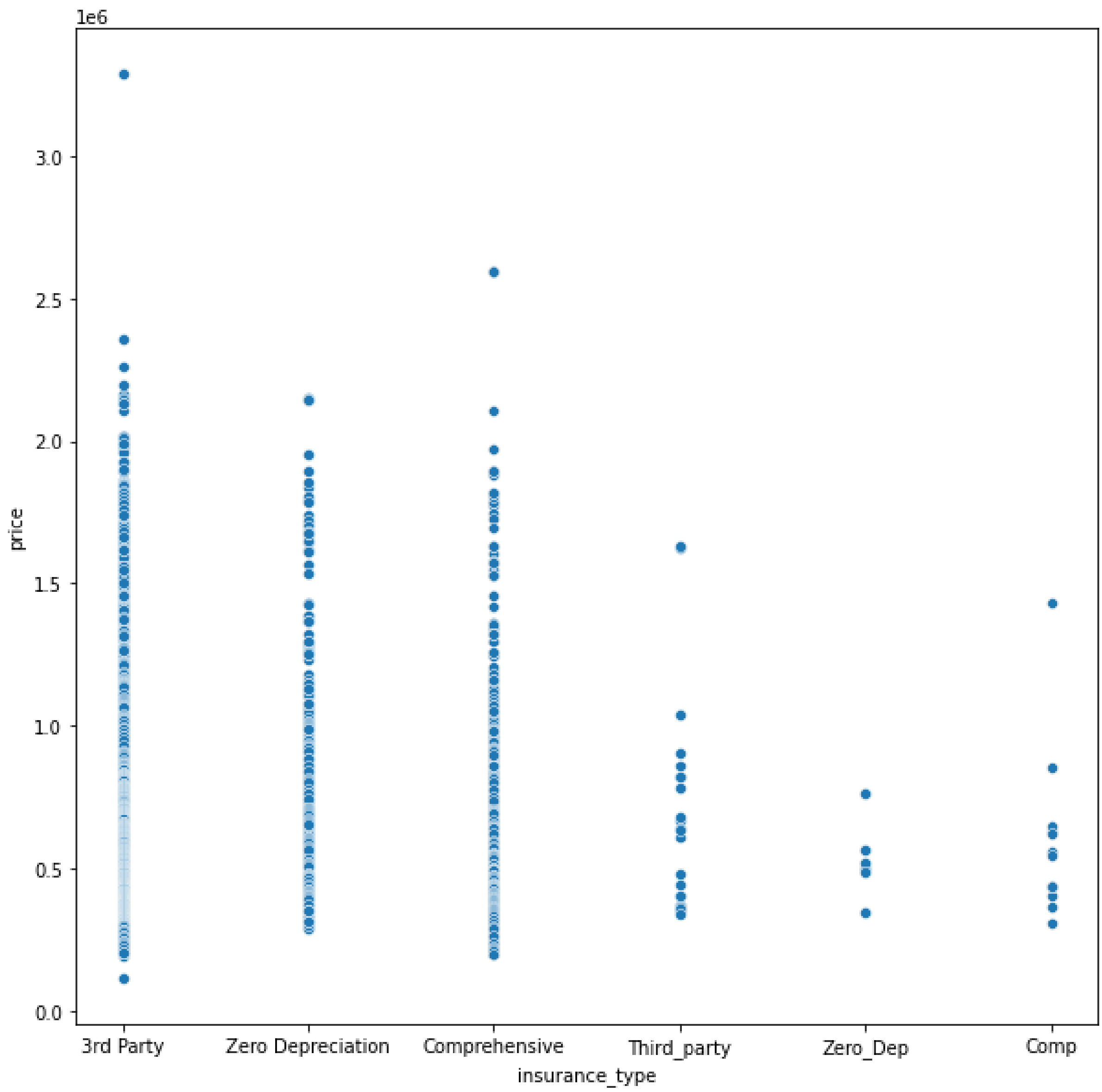
Car price is higher for Automatic Transmission cars compared to manual

Code:



```
plt.figure(figsize=(10,10))
sns.scatterplot(df['insurance_type'],df["price"])
```

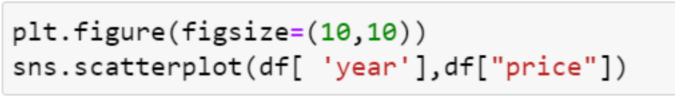
Output:



Observation:

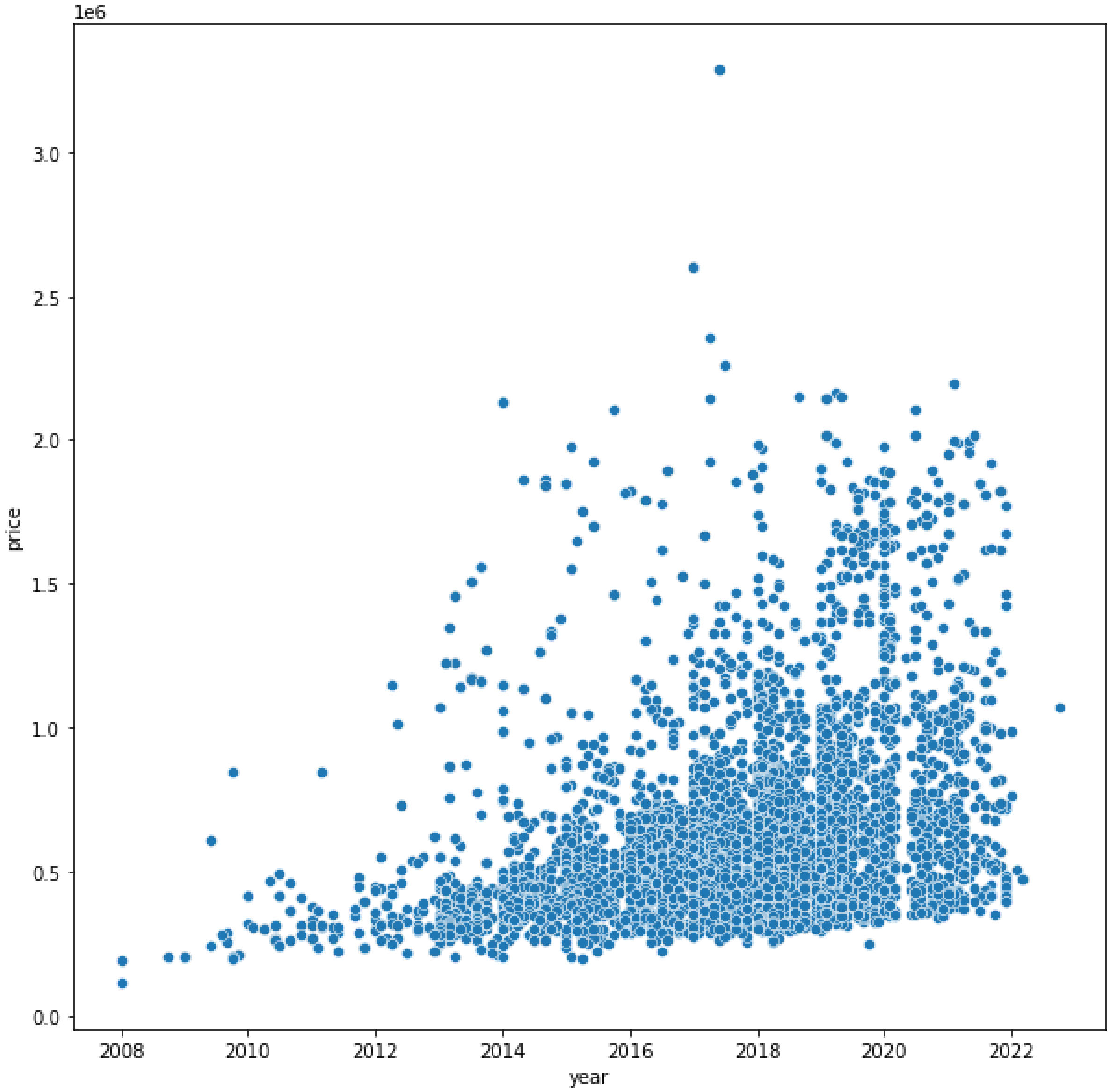
Third party insurance cars price is higher compared to others

Code:



```
plt.figure(figsize=(10,10))
sns.scatterplot(df[ 'year'],df["price"])
```

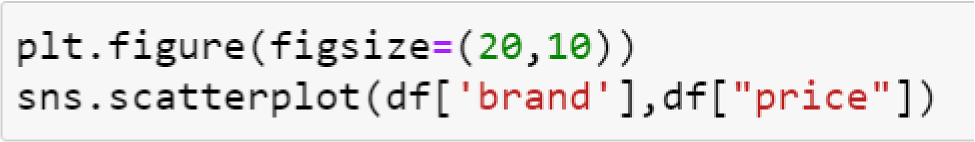
Output:



Observation:

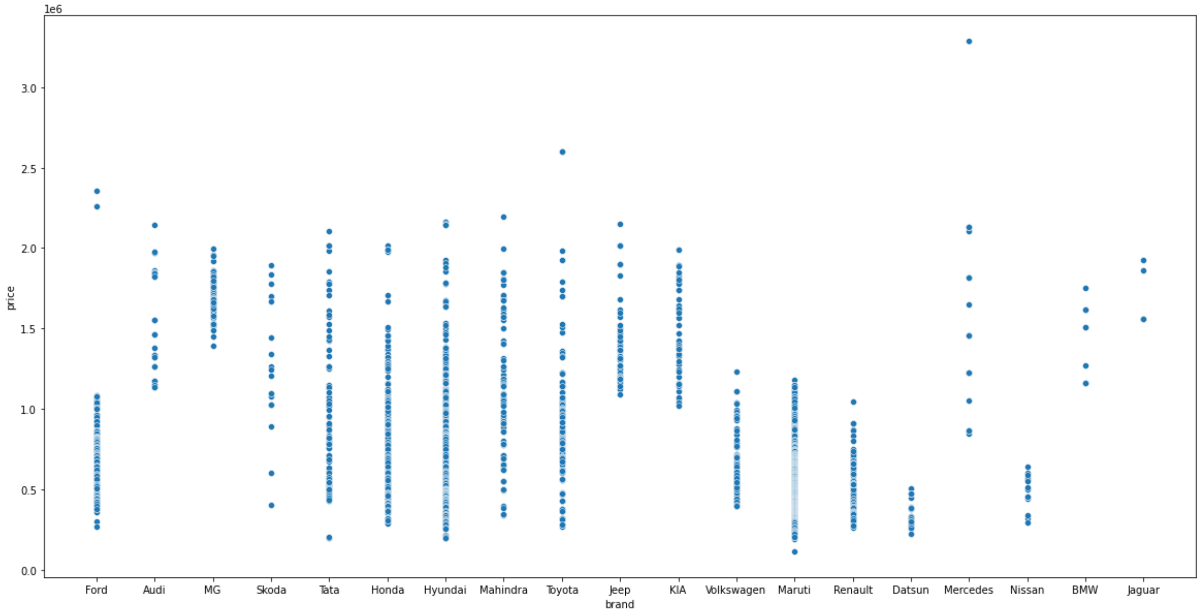
The Price of Cars increases as the year is latest.

Code:



```
plt.figure(figsize=(20,10))
sns.scatterplot(df['brand'],df["price"])
```

Output:



Observation:

The Price of Mercedes is higher compared to others
Maruti and Nissan Price is lower compared to others

- Interpretation of the Results

  Based on the visualizations I can see the Car with Newer Model the price is higher as year of car increase the price decrease, the Fuel type Diesel cars price is higher, if the last service date is latest the price also increases. For transmission auto transmission car price is higher. If the car owners increase the price decrease. And finally, if the kilometres is higher the price tends to decrease.

CONCLUSION

- Key Findings and Conclusions of the Study

We can clearly see the price of newer car’s purchased in recent years, with less kilometres, 1st owner preferably, fuel type to be diesel and with Auto transmission tends to be higher. so based on this data we can see people are ready to pay premium amount for car which is recently purchased, has latest features and with less overall kilometres covered by Car.

- Learning Outcomes of the Study in respect of Data Science

We can learn many useful insights from this study. the price of car tends to be based on kilometres, latest features, fuel type and date of purchase.

- Limitations of this work and Scope for Future Work

The Limitations are some of the Categorical data may have ordinal relationship but we considered they don’t have ordinal relationship so in future someone with in depth industry knowledge may distinguish the data and encode them