CAR PRICE PREDICTION

Contents

Chapter 1

1.1 Introduction:

Sometimes selling and buying car becomes crucial as we are not able to identify its fair price accurately. The depreciation of a car depends on a variety of factors so the car owner needs to be aware of the worth of their vehicle. With the rapid expansion of Machine Learning, this problem can also be solved by minimizing human efforts and time.

1.2 Use Cases:

1.2.1 ML can cope with price volatility

Price volatility denotes the price fluctuations of a product. To measure price volatility, you need to take a day-to-day percentage difference in a product or service's price. Through this system one can see how much price is fluctuate from past and can take decision through coming results.

1.2.2 ML models can analyze multiple data sources at once

Price prediction is hard because it requires many data sources: from internal market reports to competitors' webpages to CRM files. And it's a challenge for humans to process this volume of information, whereas, for AI, more data is nearly always better. If you pick good sources and use your data well, your software will create highly accurate predictions (that only get more accurate with time). Better still, you'll likely find patterns that you'd never spot with a manual process.

1.2.3 ML improves the accuracy of price predictions

The accuracy of traditional pricing methods leaves much to be desired. In truth, most conventional methods value intuition and subjective opinion over hard data. And that's why decisions based on such processes often lead businesses down a rabbit hole.

1.2.4 ML reduce manual efforts

ML is vast and can be more accurate through times and this process can reduce the human manual efforts and can provide accurate and precise results. Long researches and manual findings can take a lot of time if ones spend on this type of hustle.

1.2.5 ML can help you improve your profit margin

When you use machine learning, you also get an excellent grasp of how industry prices evolve over the course of a year. And this leads to a final, more subtle benefit. Suppose you spot that a supplier often increases their prices in October. You can make a note to stock up on certain goods in September, avoiding the upcoming increase, saving you money, and boosting your overall profit margin.

1.3 Dataset:

Dataset includes below columns

1. Name: Name of the model (i-e Hyundai Grand i10, Audi A8 etc)
2. Company: Model belongs to which company (i-e Hyunda, Ford etc)
3. Year: In which the model released (i-e 2009, 2016 etc)
4. Price: Price of the model (i-e 425000, 250000 etc)
5. Kms_driven: km driven would refer to how much distance (in kilometres) the car had travelled (i-e 45000, 32000 etc)
6. Fuel Type: Tell about on which fuel model works (i-e Petrol, Diesel)

```
car.head()
```

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing	Hyundai	2007	80000	45000	Petrol
1	Mahindra Jeep CL550	Mahindra	2006	425000	40	Diesel
2	Hyundai Grand i10	Hyundai	2014	325000	28000	Petrol
3	Ford EcoSport Titanium	Ford	2014	575000	36000	Diesel
4	Ford Figo	Ford	2012	175000	41000	Diesel

Chapter 2:

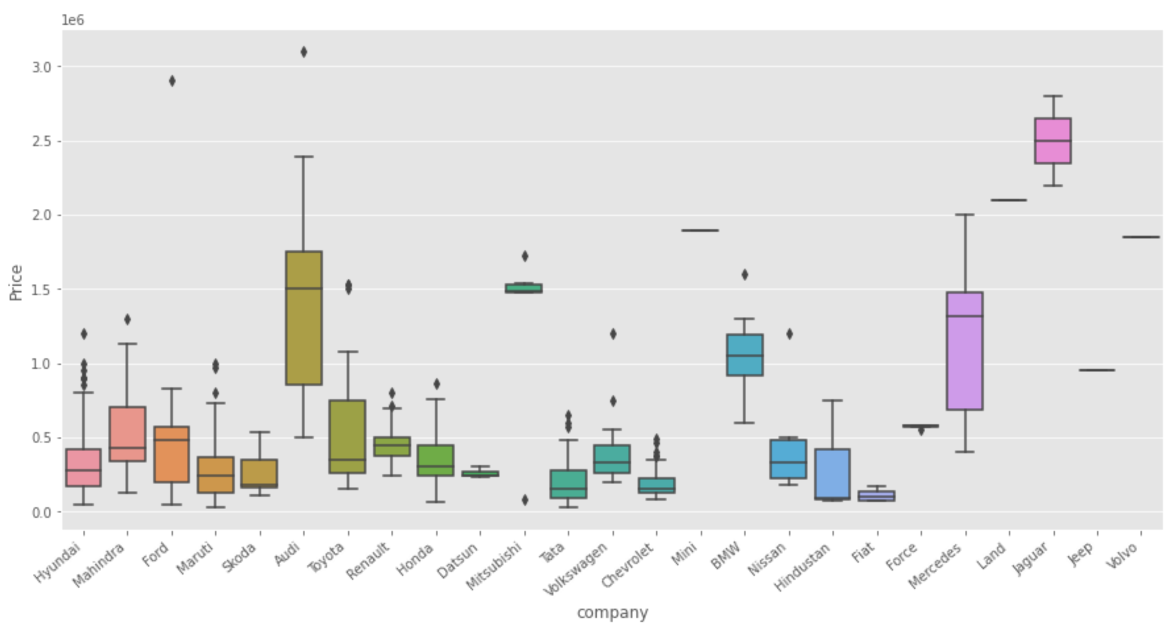
2.1 EDA:

2.1.1 Checking relationship of Company with Price

```
car['company'].unique()
```

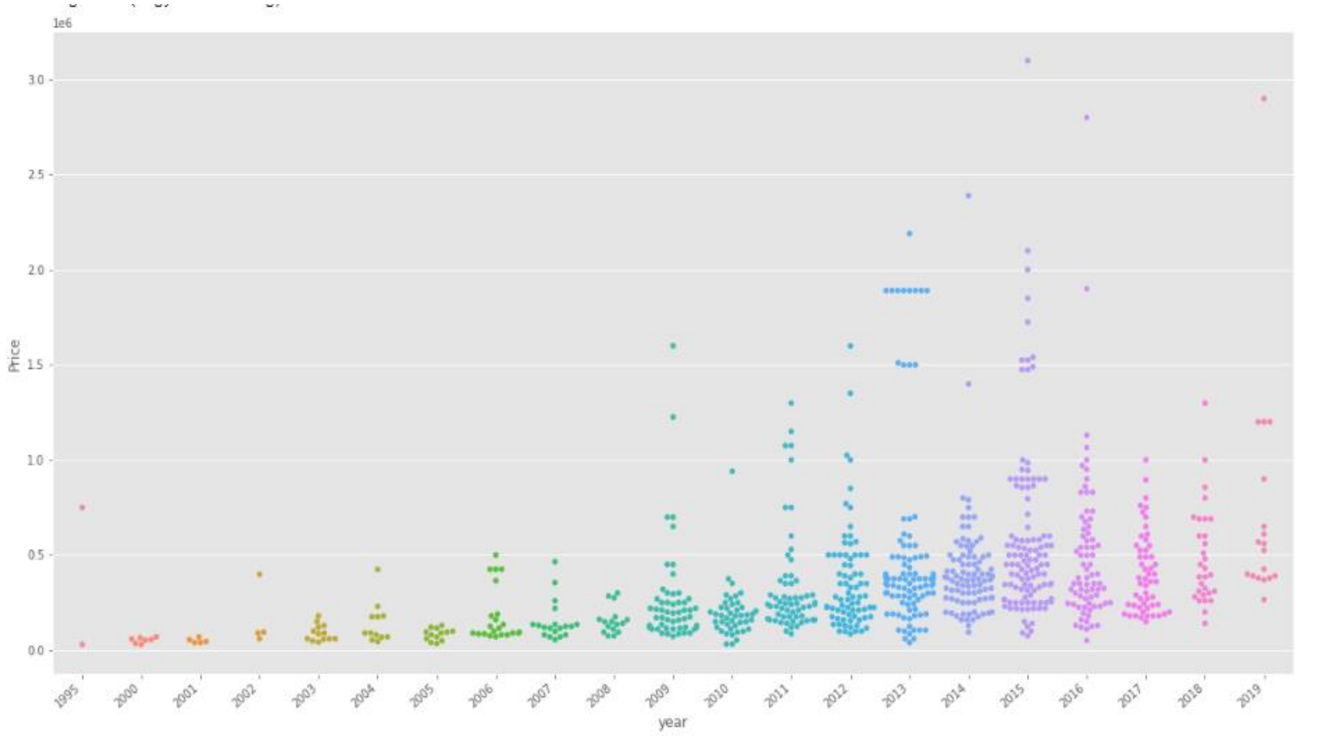
```
import seaborn as sns
```

```
plt.subplots(figsize=(15,7))
ax=sns.boxplot(x='company',y='Price',data=car)
ax.set_xticklabels(ax.get_xticklabels(),rotation=40,ha='right')
plt.show()
```



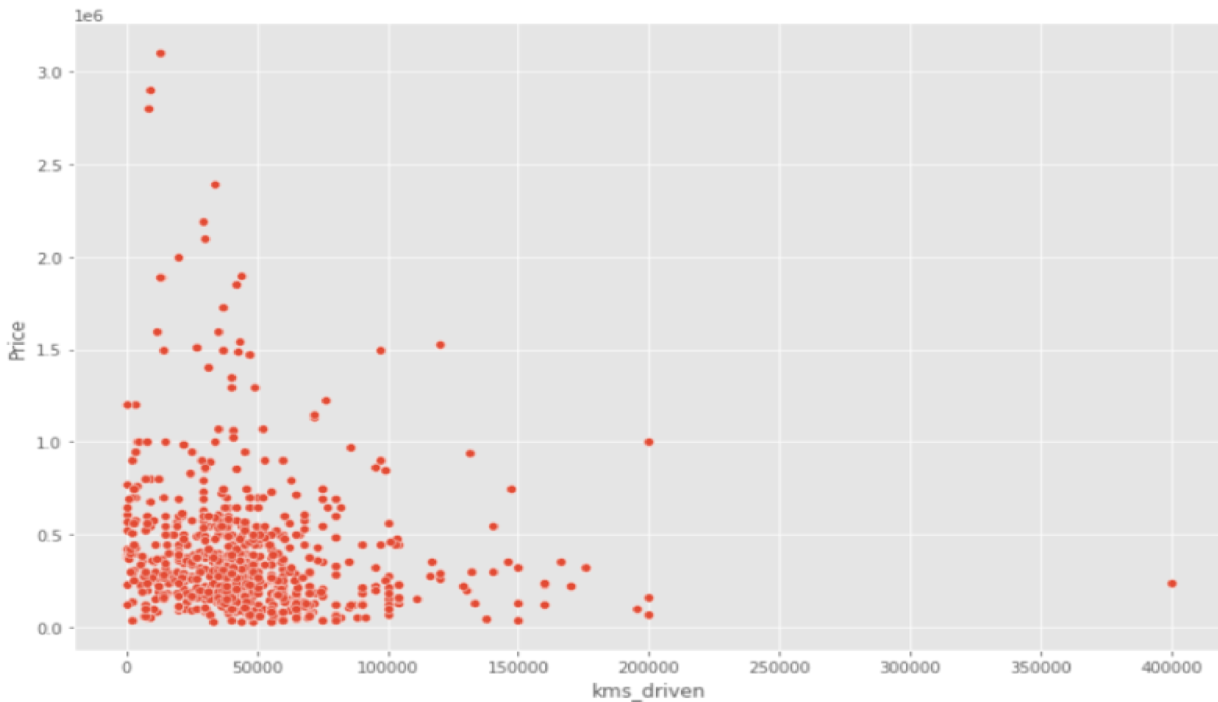
2.1.2 Checking relationship of Year with Price

```
] plt.subplots(figsize=(20,10))
  ax=sns.swarmplot(x='year',y='Price',data=car)
  ax.set_xticklabels(ax.get_xticklabels(),rotation=40,ha='right')
  plt.show()
```



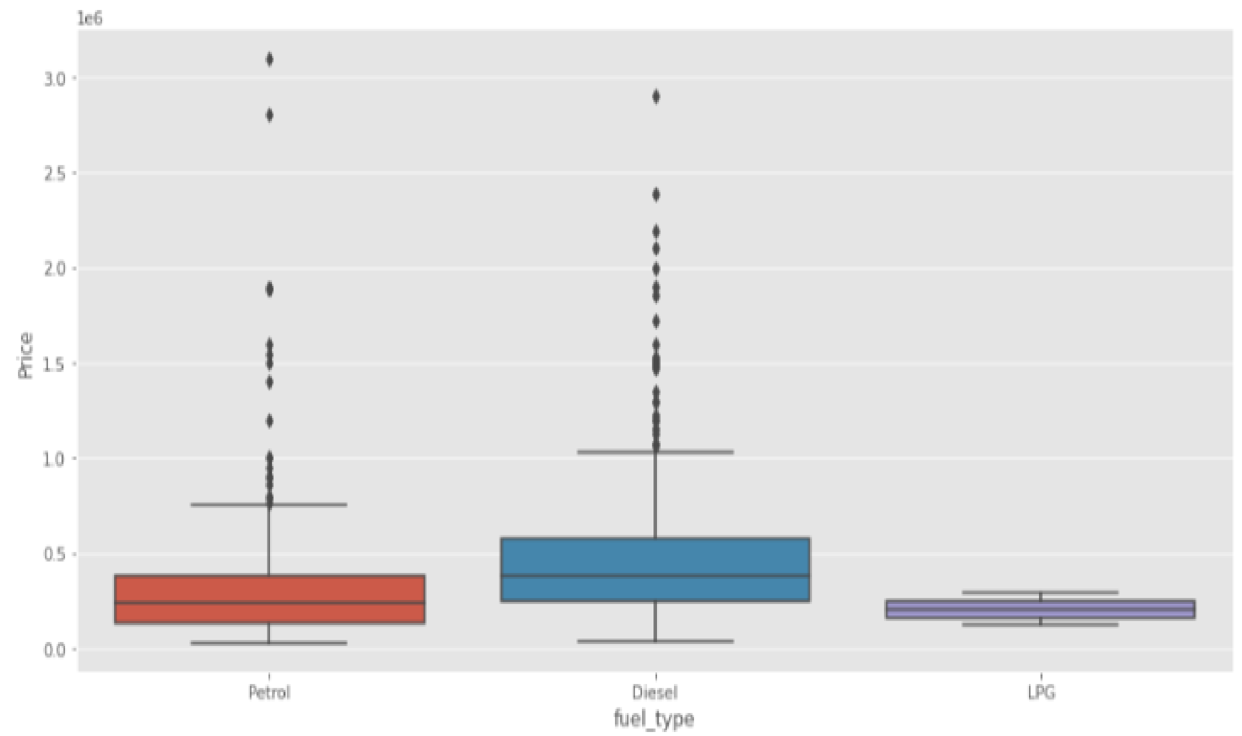
2.1.3 Checking relationship of kms_driven with Price

```
] sns.relplot(x='kms_driven',y='Price',data=car,height=7,aspect=1.5)
```



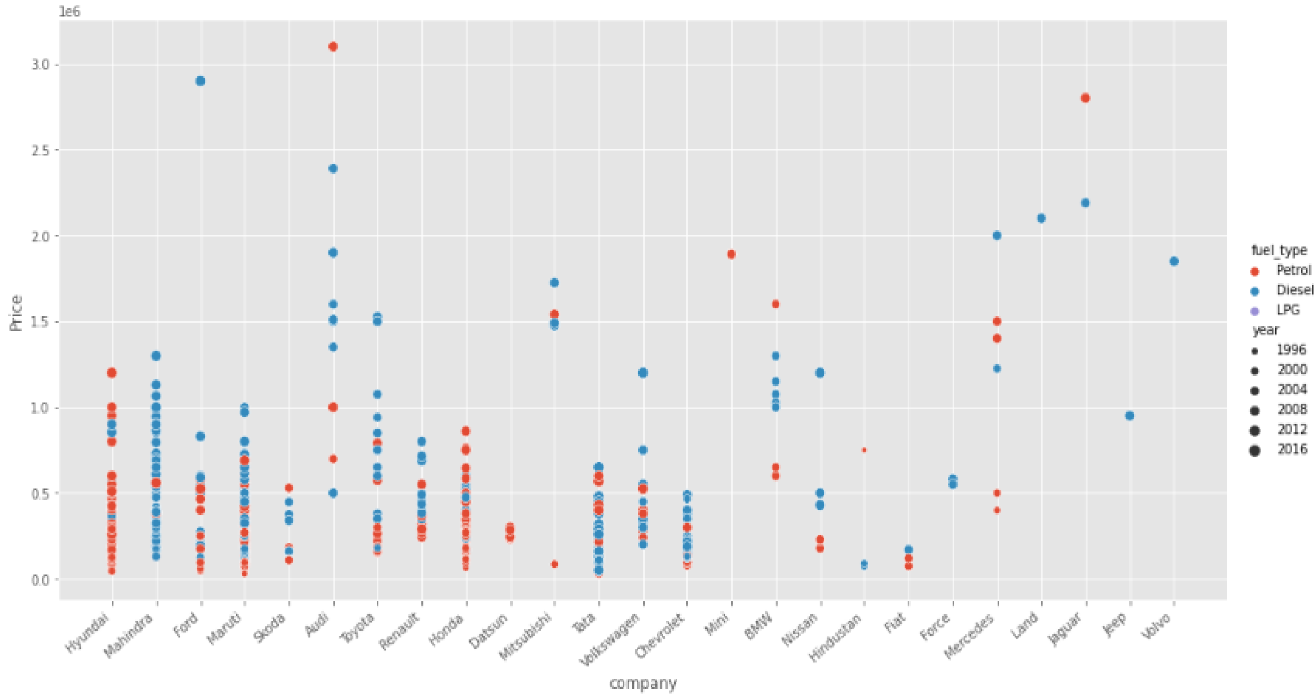
2.1.4 Checking relationship of Fuel Type with Price

```
plt.subplots(figsize=(14,7))
sns.boxplot(x='fuel_type',y='Price',data=car)
```



2.1.5 Relationship of Price with FuelType, Year and Company mixed

```
ax=sns.relplot(x='company',y='Price',data=car,hue='fuel_type',size='year',height=7,aspect=2)
ax.set_xticklabels(rotation=40,ha='right')
```



1e6
3.0
2.5
2.0
1.5
1.0
0.5
0.0
Price
Hyundai
Mahindra
Ford
Maruti
Skoda
Audi
Toyota
Renault
Honda
Datsun
Mitsubishi
Tata
Volkswagen
Chevrolet
Mini
BMW
Nissan
Hindustan
Fiat
Force
Mercedes
Land
Jaguar
Jeep
Volvo
company
fuel_type
Petrol
Diesel
LPG
year
1996
2000
2004
2008
2012
2016

2.2 Model:

Linear Regression was chosen as the model due to its simplicity and comparatively small training time. The features, without any feature mapping, were used directly as the feature vectors.

2.2.1 Extracting Training Data

```
[33] X=car[['name','company','year','kms_driven','fuel_type']]
     y=car['Price']
```

```
[34] X
```

```
[35] y.shape
```

2.2.2 Applying Train Test Split

```
[36] from sklearn.model_selection import train_test_split
     X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2)
```

```
[37] from sklearn.linear_model import LinearRegression
```

```
[38] from sklearn.preprocessing import OneHotEncoder
     from sklearn.compose import make_column_transformer
     from sklearn.pipeline import make_pipeline
     from sklearn.metrics import r2_score
```

2.2.3 Linear Regression Model

```
[41] lr=LinearRegression()
```

Making a pipeline

```
[42] pipe=make_pipeline(column_trans,lr)
```

Fitting the model

```
[43] pipe.fit(X_train,y_train)
```

```
[44] y_pred=pipe.predict(X_test)
```

Checking R2 Score

```
[45] r2_score(y_test,y_pred)
```

Finding the model with a random state of TrainTestSplit where the model was found to give almost 0.92 as r2_score

```
[72] scores=[]
     for i in range(1000):
         X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.1,random_state=i)
         lr=LinearRegression()
         pipe=make_pipeline(column_trans,lr)
         pipe.fit(X_train,y_train)
         y_pred=pipe.predict(X_test)
         scores.append(r2_score(y_test,y_pred))
```

```
[73] np.argmax(scores)

655
```

```
[74] scores[np.argmax(scores)]

0.920087093218515
```

```
pipe.predict(pd.DataFrame(columns=X_test.columns,data=np.array(['Maruti Suzuki Swift','Maruti',2019,100,'Petrol']).reshape(1,5)))

array([400642.51767152])
```

The best model is found at a certain random state

```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.1,random_state=np.argmax(scores))
lr=LinearRegression()
pipe=make_pipeline(column_trans,lr)
pipe.fit(X_train,y_train)
y_pred=pipe.predict(X_test)
r2_score(y_test,y_pred)

0.920087093218515
```

Chapter 3

3.1 Conclusion

Car price prediction can be a challenging task due to the high number of attributes that should be considered for the accurate prediction. The major step in the prediction process is collection and preprocessing of the data. Data cleaning is one of the processes that increases prediction performance, yet insufficient for the cases of complex data sets as the one in this research. Therefore, there is an urgent need for a 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.