Car Price Prediction

Discover the factors that influence car prices and how to build a model to model to predict them. Take a deep dive into the exciting world of automotive automotive data science.

by Anil Poddar

Introduction

Welcome to the Car Price Prediction project! In this endeavor, we leverage the power of machine learning to forecast car prices based on a myriad of features. Whether you're a car enthusiast, a data scientist, or someone navigating the automotive market, this project aims to provide valuable insights and predictions.

Title:

1: Data Exploration

2: Data cleaning & pre_processing

3: Factor influencing car prices

4: Data collection & pre-processing

5: Exploratory data analysis

6: EDA for categorical column

7: EDA for continuous column

8: BOX plot

9: Pair Plot

10: Correlation plot

11: Encoding technique

12: Feature selections and model evaluation

13: Model selection & training

14: Model evaluation & validation

15: Hyper parameter tuning

16: Application of the price production model

17: Web application

18: Conclusion & Future Work

Data Exploration

This data is collected from 'CAR DETAILS'.

Following details of cars are included in the dataset:

1) Car name 2) Year 3) Selling Price 4) Kms driven 5) Fuel 6) Seller type 7) Transmission 8) Owner

In [1]:
```
import numpy as np # perform a wide variety of mathematitical oprations on array
import pandas as pd  # data preprocessing
import seaborn as sns  # visualization
import matplotlib.pyplot as plt  # visualization
from sklearn import datasets  #implement machine learning models and statistical modelling
import warnings
warnings.filterwarnings('ignore')
```

Load Data Set

In [2]:
```
#Import Data
df = pd.read_csv('CAR DETAILS.csv')
```

1.Display top 5 rows of the Dataset. ¶

In [3]:
```
df.head()
```

Out[3]:

	name	year	selling_price	km_driven	fuel	seller_type	transmission	owner
0	Maruti 800 AC	2007	60000	70000	Petrol	Individual	Manual	First Owner
1	Maruti Wagon R LXI Minor	2007	135000	50000	Petrol	Individual	Manual	First Owner
2	Hyundai Verna 1.6 SX	2012	600000	100000	Diesel	Individual	Manual	First Owner
3	Datsun RediGO T Option	2017	250000	46000	Petrol	Individual	Manual	First Owner
4	Honda Amaze VX i-DTEC	2014	450000	141000	Diesel	Individual	Manual	Second Owner

Data Cleaning & Pre-proccessings

Duplicates

```
df.duplicated().sum()
```

```
763
```

```
df.drop_duplicates(inplace=True)
df.shape
```

```
(3577, 8)
```

```
df.dtypes
```

```
name             object
year              int64
selling_price     int64
km_driven         int64
fuel             object
seller_type      object
transmission     object
owner            object
dtype: object
```

Checking Duplicates and Drop Drop it

5.Check null values in the Dataset

```
df.isnull()
```

	name	year	selling_price	km_driven	fuel	seller_type	transmission	owner
0	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...
4335	False	False	False	False	False	False	False	False
4336	False	False	False	False	False	False	False	False
4337	False	False	False	False	False	False	False	False
4338	False	False	False	False	False	False	False	False
4339	False	False	False	False	False	False	False	False

4340 rows × 8 columns

Find null values and remove it to clean a clean a data

Data Cleaning & Pre-proccessings

```
df['brand']=brand
df['model']=model
df['sub_class']=sub_class
df.head()
```

	name	year	selling_price	km_driven	fuel	seller_type	transmission	owner	brand	model	sub_class
0	Maruti 800 AC	2007	60000	70000	Petrol	Individual	Manual	First Owner	Maruti	800	AC
1	Maruti Wagon R LXI Minor	2007	135000	50000	Petrol	Individual	Manual	First Owner	Maruti	Wagon	R LXI Minor
2	Hyundai Verna 1.6 SX	2012	600000	100000	Diesel	Individual	Manual	First Owner	Hyundai	Verna	1.6 SX
3	Datsun RediGO T Option	2017	250000	46000	Petrol	Individual	Manual	First Owner	Datsun	RediGO	T Option
4	Honda Amaze VX i-DTEC	2014	450000	141000	Diesel	Individual	Manual	Second Owner	Honda	Amaze	VX i-DTEC

Approach

- We have 1497 unique features in name column
- To Reduce the complexity in name column we we have created three new column such brand, brand, model, sub_class
- we can drop the name column

```
car_names=list(df['name'])

brand,model,sub_class=[],[],[]

for car in car_names:

parts=car.split()

x=parts[0]

y=parts[1]

z=parts[2:]

brand.append(x)

model.append(y)

sub_class.append(z)
```

Factors Influencing Car Prices

1. **Brand**

   The reputation of a car brand and how people perceive it affects the price.

2. **Year, Make, & Model**

   The year, make, and model of a car determine the base price of a car.

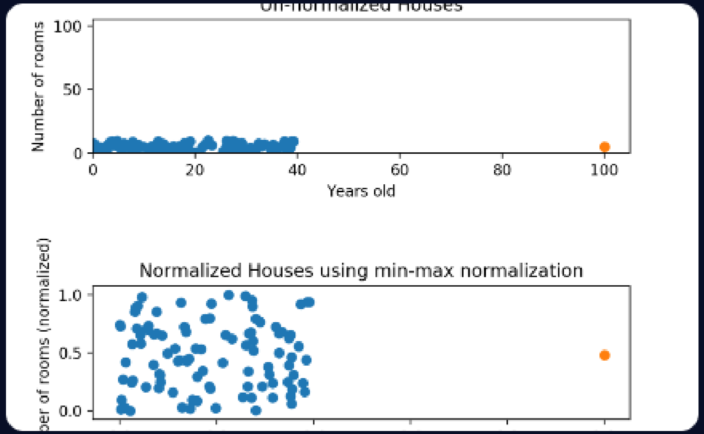
3. **Mileage**

   The more mileage on a car, the lower its price price will be.

4. **Condition**

   The physical and mechanical condition of a car of a car plays a crucial role in its value.

Data Collection and Preprocessing



Data Collection

Exploring various sources to collect structured and unstructured data.

Data Cleaning

Removing outliers, filling missing values, and transforming data.

Data Normalization

Scaling and standardizing data to data to eliminate bias.

Exploratory Data Analysis

Univariate Analysis

Study the data distributions

Bivariate Analysis

Find correlations between independent and and dependent variables

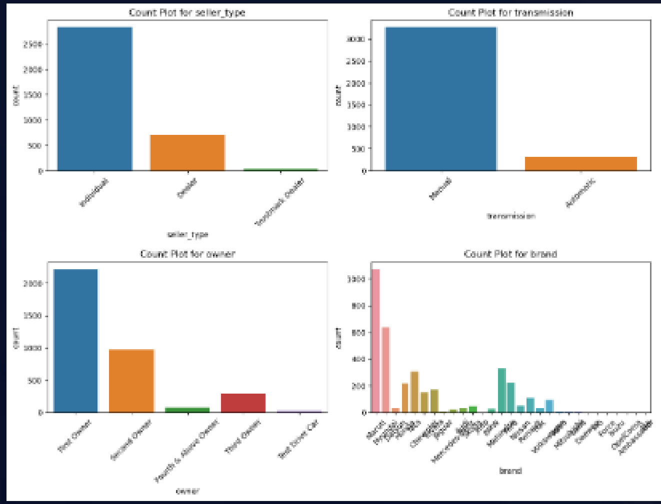
Multivariate Analysis

Understand the complicated relationships between variables

Data Visualization

Present data visually using charts and graphs graphs

EDA for Categorical Columns




```
plt.figure(figsize=(12,12))

for i, cat in enumerate(cat_cols[1:5]):
:

plt.subplot(3,2,i+1)

sns.countplot(data=df,x=cat=cat)

plt.xticks(rotation=45)

plt.title(f'Count Plot for {cat}')

plt.tight_layout()

#plt.savefig("Categorical Plot")

plt.show()
```

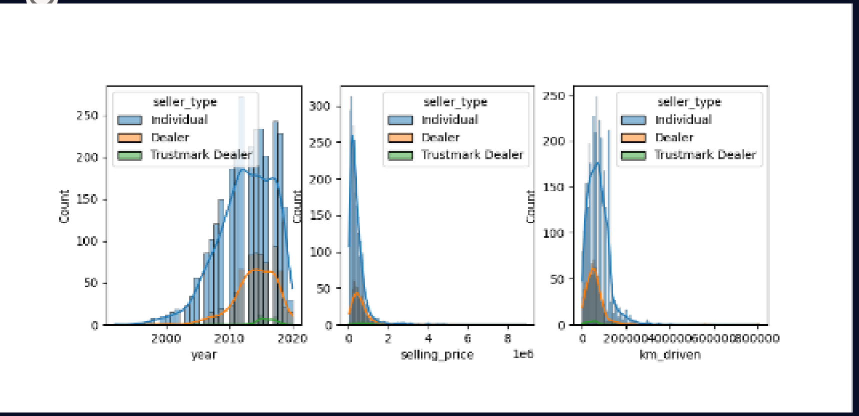
Insights

- Most of the used cars available are belongs to petrol and Diesel category
- All most all the cars transmisson are of Manually operated
- Available cars are more likely to be First and 2nd Hand users only
- Sellers prefer to sell their car directly to customers (individual) compare to Dealer or Trusted Dealer
- Maruti, Hyundai, Mahindra, Tata are most of used cars available in market compare to other brands of cars

EDA for Continous Columns

Insights

- Most of the available used used cars released after after 2005
- 90 percentage of cars are sold for less than 1500000
- Sellers prefers to sell their cars before reaching 200000 km
- Its clearly indicates individual sellers are more available in the market compare to Third party sellers



plt.figure(figsize=(10,8))

for i in range(len(num_cols)): range(len(num_cols)):

plt.subplot(2,3,i+1)

sns.histplot(data=df,x=num_cols[i],kde=True,hue='seller_type')

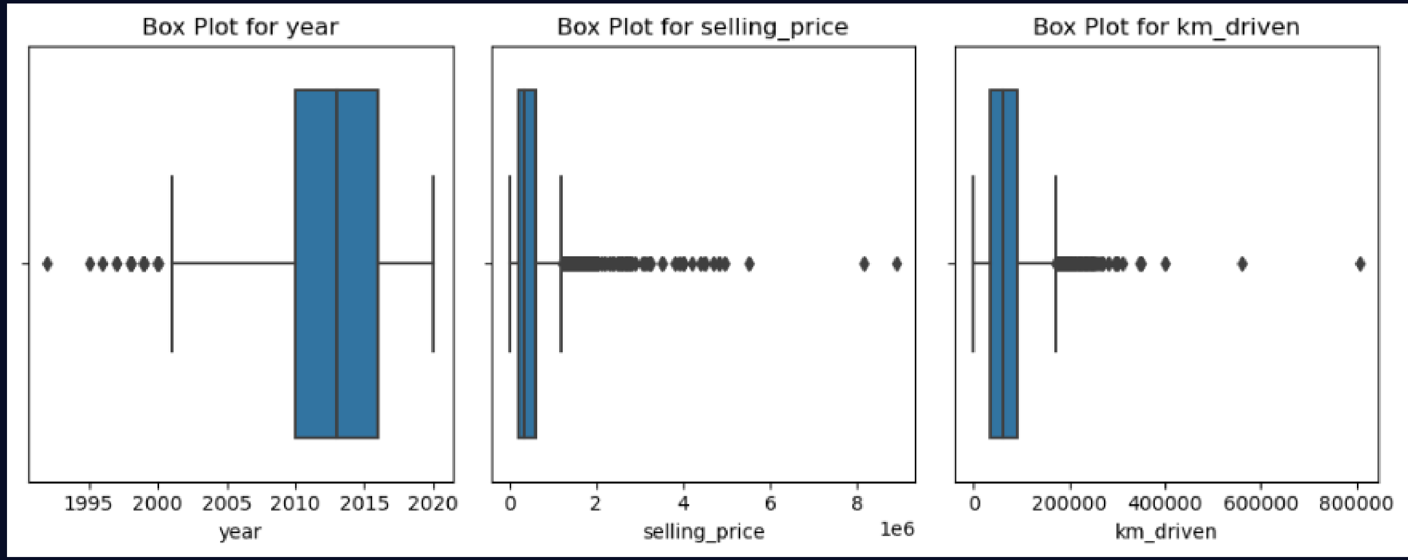
plt.xticks(rotation=0)

plt.savefig('Histoplot ')

plt.tight_layout()

plt.show()

Box Plot

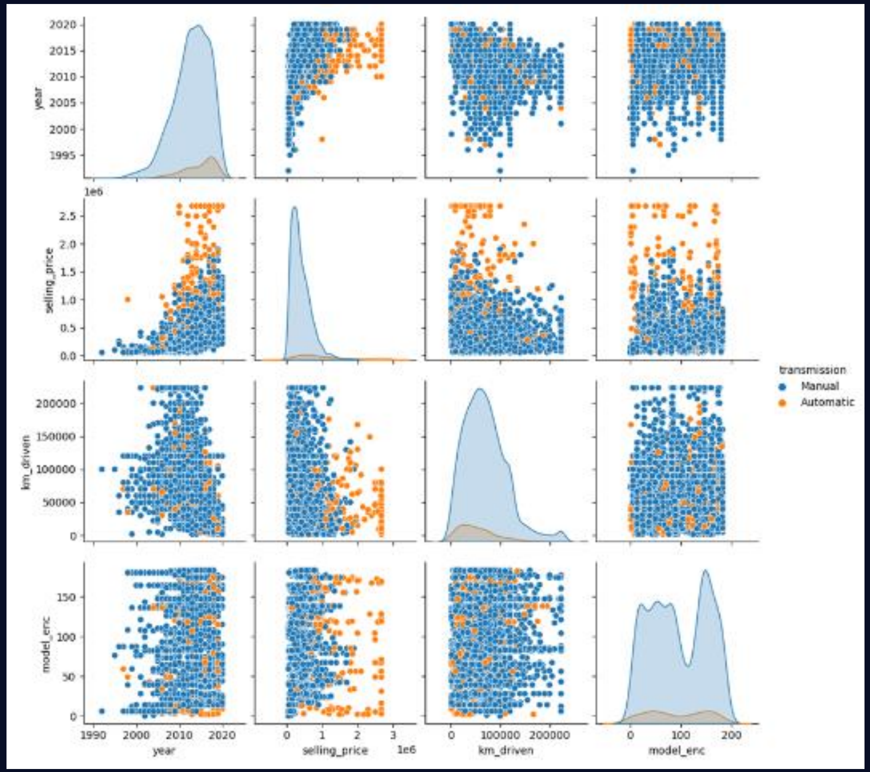


#Treating the selling and Km_driven columns

```
df['selling_price']=np.where(df["selling_price"]>2675000.0,2675000.0,df['selling_price'])

df['km_driven']=np.where(df['km_driven']>223158.4,223158.4,df['km_driven'])

df['selling_price']=np.where(df["selling_price"]<51786.64,51786.64,df['selling_price'])

df['km_driven']=np.where(df['km_driven']<1744.08,1744.08,df['km_driven'])
```

Pair Plot

```
sns.pairplot(df,hue='transmission')

plt.savefig("Paiplot for continuous variable")

plt.show()
```



Insights

Plot Selling Price vs Year

- Recently released model are costlier than than old models
- price of the automatic versions appears to costiler compare to manual transmission

plot year vs Km_driven

- There is no partical relationship between model Realeased Year with respect to KM of vehicle
- manual and automatic transmission dont seems to some category

Selling price vs KM_driven

- Lower the Km driven Higher the cost of vehicle
- automatic transmission seems to be costiler compare to manual transmission

Correlation Plot

```
df.columns
```

```
Index(['year', 'selling_price', 'km_driven', 'fuel', 'seller_type',
       'transmission', 'owner', 'brand', 'Age', 'model_enc'],
      dtype='object')
```

```
df_encoded.columns
```

```
Index(['year', 'selling_price', 'km_driven', 'fuel', 'seller_type',
       'transmission', 'Age', 'model_enc', 'owner_First Owner',
       'owner_Fourth & Above Owner', 'owner_Second Owner',
       'owner_Test Drive Car', 'owner_Third Owner', 'brand_Ambassador',
       'brand_Audi', 'brand_BMW', 'brand_Chevrolet', 'brand_Daewoo',
       'brand_Datsun', 'brand_Fiat', 'brand_Force', 'brand_Ford',
       'brand_Honda', 'brand_Hyundai', 'brand_Isuzu', 'brand_Jaguar',
       'brand_Jeep', 'brand_Kia', 'brand_Land', 'brand_MG', 'brand_Mahindra',
       'brand_Maruti', 'brand_Mercedes-Benz', 'brand_Mitsubishi',
       'brand_Nissan', 'brand_OpelCorsa', 'brand_Renault', 'brand_Skoda',
       'brand_Tata', 'brand_Toyota', 'brand_Volkswagen', 'brand_Volvo'],
      dtype='object')
```

<u>One Hot Encoding</u>

df_encoded=pd.get_dummies(df)

df_encoded.head()

corr=df[num_cols].corr()

sns.heatmap(corr,cmap='coolwarm',annot=True)

plt.savefig('Correlation Graph')

plt.show()

Encoding Technices

<u>Label Encoding</u>

```
from sklearn.preprocessing import LabelEncoder

lb=LabelEncoder()

df['model_enc']=lb.fit_transform(df['model'])
```

Feature selections and Model evaluation evaluation

Feature Slection

```
x=df_encoded.drop('selling_price',axis=1)

y=df_encoded['selling_price']

print(x.shape)

print(y.shape)
```

Model Evaluation

```
def model_eval(x_train,x_test,y_train,y_test,model,mname):


model.fit(x_train,y_train)

ypred=model.predict(x_test)

mae=mean_absolute_error(y_test,ypred)

mse=mean_squared_error(y_test,ypred)

rmse=np.sqrt(mse)

train_scr=model.score(x_train,y_train)

test_scr=model.score(x_test,y_test)

res=pd.DataFrame({"Train_scr":train_scr,"Test_scr":test_scr,'RMS
E':rmse,'MSE':mse,

"MAE":mae},index=[mname])

return res
```

Model Selection and Training

1 Feature Selection

Select the best features that impact the target variable the most

2 Model Selection

Choose a suitable model based on the dataset's characteristics

3 Hyperparameter Tuning

Optimize the model's performance by fine-tuning its parameters

Model Selection and Training

Model Selection.

Choose the appropriate machine machine learning algorithms, such such as Linear Regression, Decision Decision Tree, Random forest, bagging etc.

Model Training

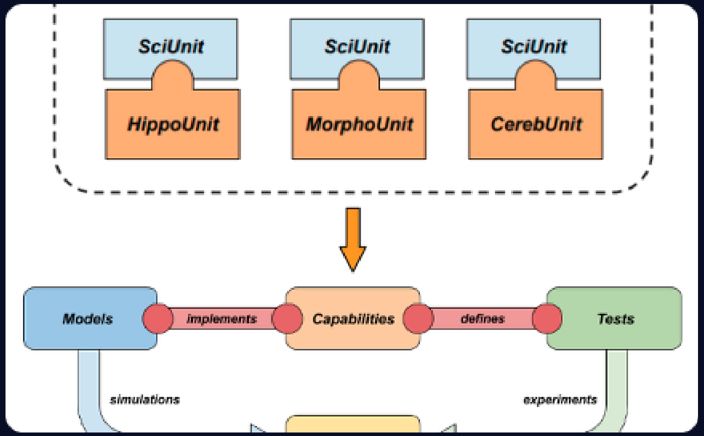
Train the selected models on the preprocessed data, optimizing hyperparameters and evaluating performance.

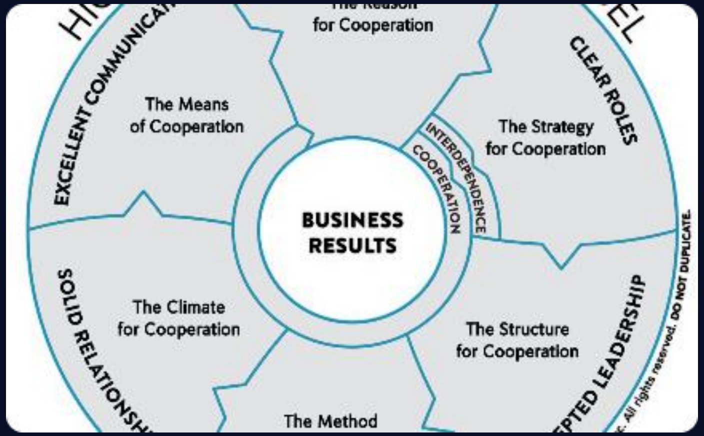
	Train_scr	Test_scr	RMSE	MSE	MAE
Linear	0.722	0.676	238241.229	5.675888e+10	159968.298
Ridge	0.721	0.676	238230.370	5.675371e+10	160653.456
Lasso	0.721	0.676	238230.370	5.675371e+10	160653.456
Decision Tree	0.874	0.785	193943.168	3.761395e+10	123314.827
Bagging for DT	0.874	0.785	193943.168	3.761395e+10	123314.827
AddaBoost for DT	0.874	0.785	193943.168	3.761395e+10	123314.827
Random Forest	0.874	0.785	193943.168	3.761395e+10	123314.827
Bagging for RF	0.874	0.785	193943.168	3.761395e+10	123314.827
AdaBoost for RF	0.932	0.826	174283.030	3.037457e+10	106065.855
KNeighbours	0.385	-0.047	428018.135	1.831995e+11	257820.253
Bagging for KN	0.385	-0.047	428018.135	1.831995e+11	257820.253
AdaBoost for KN	0.385	-0.047	428018.135	1.831995e+11	257820.253

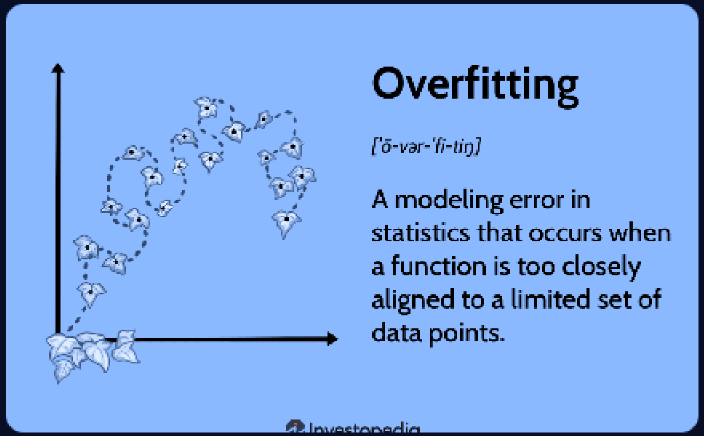
Insights

- All model suffers high over fitting and high error
- Tree model appears to more effective compare linear models
- KNeighbours appears to be less effective and errors are also high which indicated indicated data is not classified(groups)

Model Evaluation and Validation







Validation Strategy

Split the dataset into training and testing sets. Check the model's performance on it.

Performance Metrics

Use metrics like MSE, RMSE, MAE, MAE, and R-squared to evaluate evaluate the model.

Overfitting Detection and and Correction

Prevent the model from becoming too complex or fitting the data too closely.

Hyper parameter tuning

```python
from sklearn.model_selection import GridSearchCV
# Define the hyperparameter grid to search
param_grid = {
    'n_estimators': [50, 100, 200],
    'max_depth': [ 10,15,20],
    'min_samples_split': [2, 5, 10,15],
    'min_samples_leaf': [1, 2, 4]
}
# Create the GridSearchCV object
grid_search = GridSearchCV(rf, param_grid, cv=5)
# Fit the grid search to the data
grid_search.fit(x_train, y_train)

# Print the best hyperparameters
print("Best hyperparameters found:")
print(grid_search.best_params_)

# Evaluate the model with the best hyperparameters on the test set
best_rf_model = grid_search.best_estimator_
accuracy = best_rf_model.score(x_test, y_test)
print(f"Accuracy on the test set: {accuracy:.2f}")
```

```
Best hyperparameters found:
{'max_depth': 20, 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 50}
Accuracy on the test set: 0.84
```

Choosing the Best model

```
import pickle
```

```
final_model=RandomForestRegressor(n_estimators=200,max_depth=15,min_samples_split=5,min_samples_leaf=1)
final_model.fit(x_train,y_train)
```

```
RandomForestRegressor
RandomForestRegressor(max_depth=15, min_samples_split=5, n_estimators=200)
```

```
pickle.dump(final_model,open('final.pkl','wb'))
```

1 Input Data

Provide the significant columns for machine learning model after data-preprocessing

2 Model Prediction

Utilize the selected model to predict the selling price of the used car accurately.

3 Web Application

By using stream lit Develop web application so that prediction can easily accessible to everyone without the use of original data set

Application of the Price Prediction Model

Real-time Prediction

With a trained model, you can get an accurate price prediction for any car in seconds

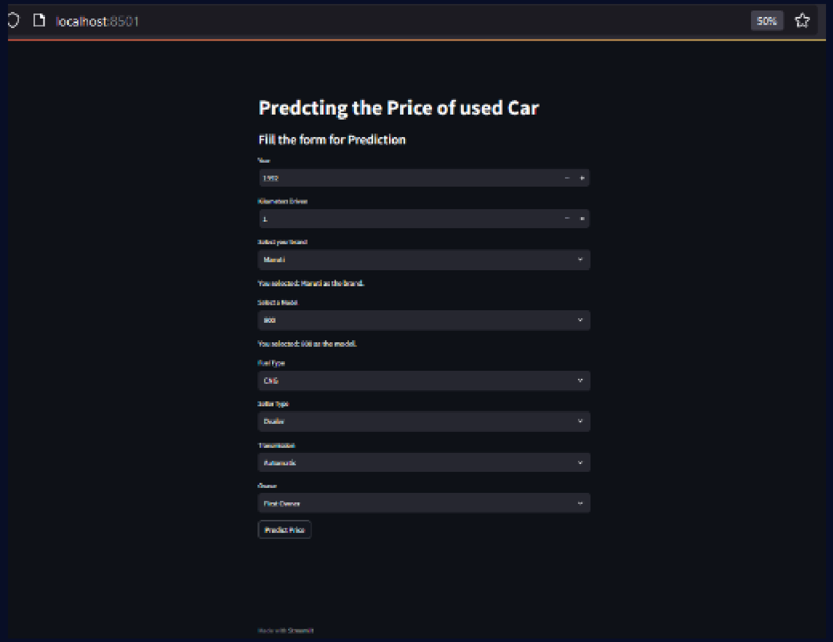
Consumer Price Awareness

Helps buyers and sellers understand the factors affecting car prices

Automation

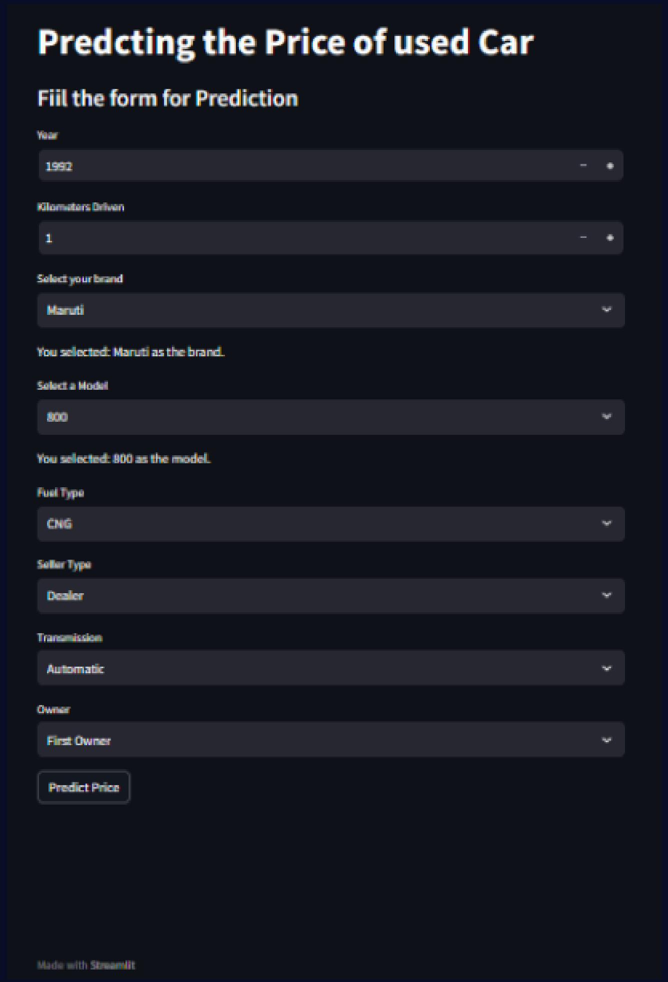
You can use the model to automate the pricing process if you're in the business of buying and selling cars

Web Application



Application Before Deployment

It run at local host URL : http://localhost:8501



After Deployment Of Application

LINK URL of web app: https://car-price-prediction-ojsqx4h4pw6ubdubmbzjxr.streamlit.app/

Conclusion and Future Work

1 Conclusion

Building a predictor model for car prices is a powerful tool that has many applications in the applications in the automotive industry.

2 Future Work

Apply machine learning techniques like deep learning and neural networks for more more accurate predictions or improving the model's performance for better results. results.

Thank you….

Thank you for joining us today for this presentation. We appreciate your time and attention as we dive into the dive into the topic at hand.

In this deck, we will explore various aspects related to car price prediction. We will take you through the journey of the journey of understanding the factors influencing car prices, data exploration and cleaning, as well as the well as the application of a price prediction model.

Throughout this presentation, we will provide insights and analysis to help you gain a deeper understanding of the understanding of the car pricing domain. We hope that the information shared here will be valuable to you and to you and provide a foundation for your own exploration and research.

Once again, thank you for being a part of this presentation. Let's get started!

Prediction Link:https://car-price-prediction-ojsqx4h4pw6ubdubmbzjxr.streamlit.app/