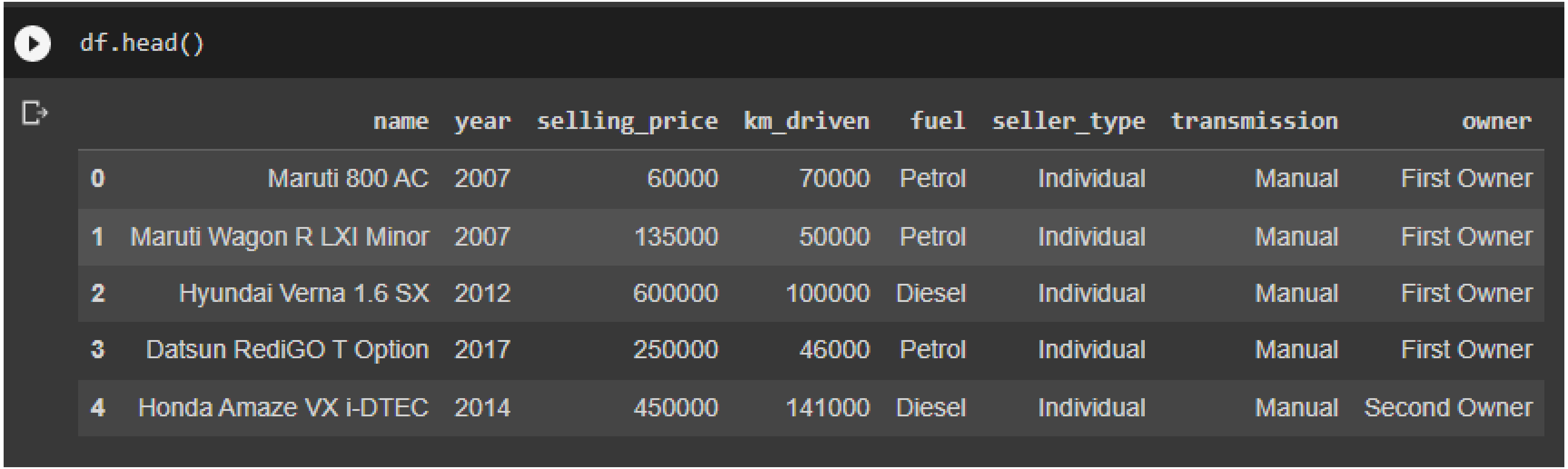
Data Science with Python Career Program - Capstone Project

- By Omkar Barge

Agenda

- Data Exploration
- Data insights
- EDA Graphs.
- Graphical Analysis and conclusion on Data
- Data Cleaning & Pre-Processing Steps.
- ML Modeling
- Deployment of ML Models using Streamlit.

Data Exploration

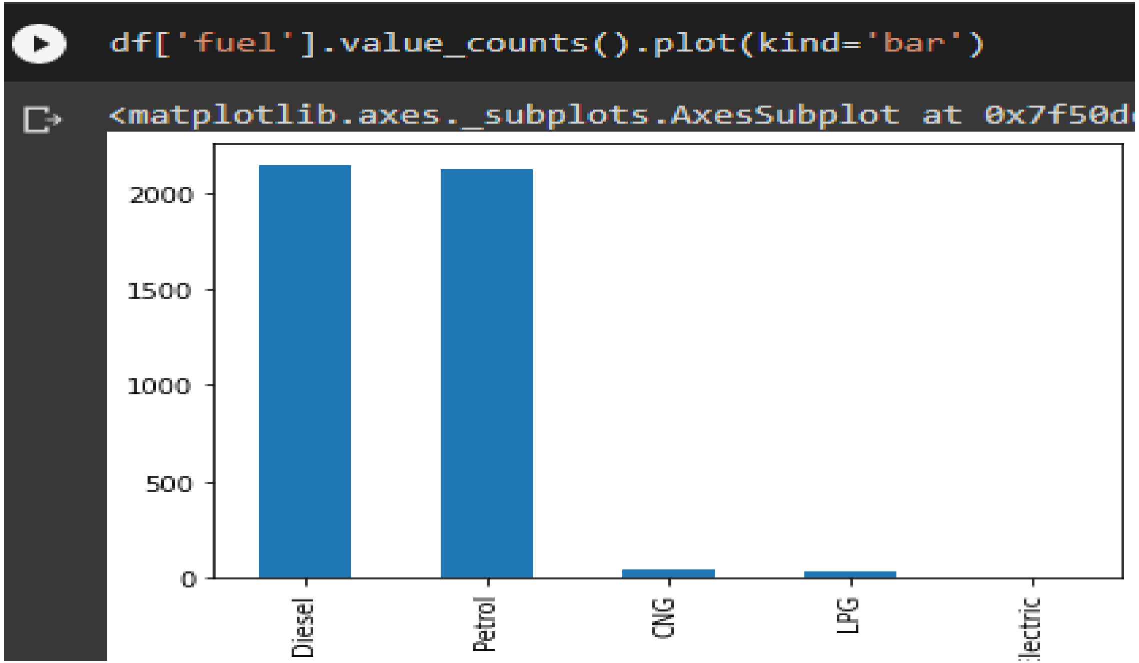


```
df.head()
```

	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

Dataset contains information of used cars.
Information present in dataset is Name of company, Manufactured Year, Selling Price, Km driven, Fuel, Seller Type, Transmission, Owner.

Data insights

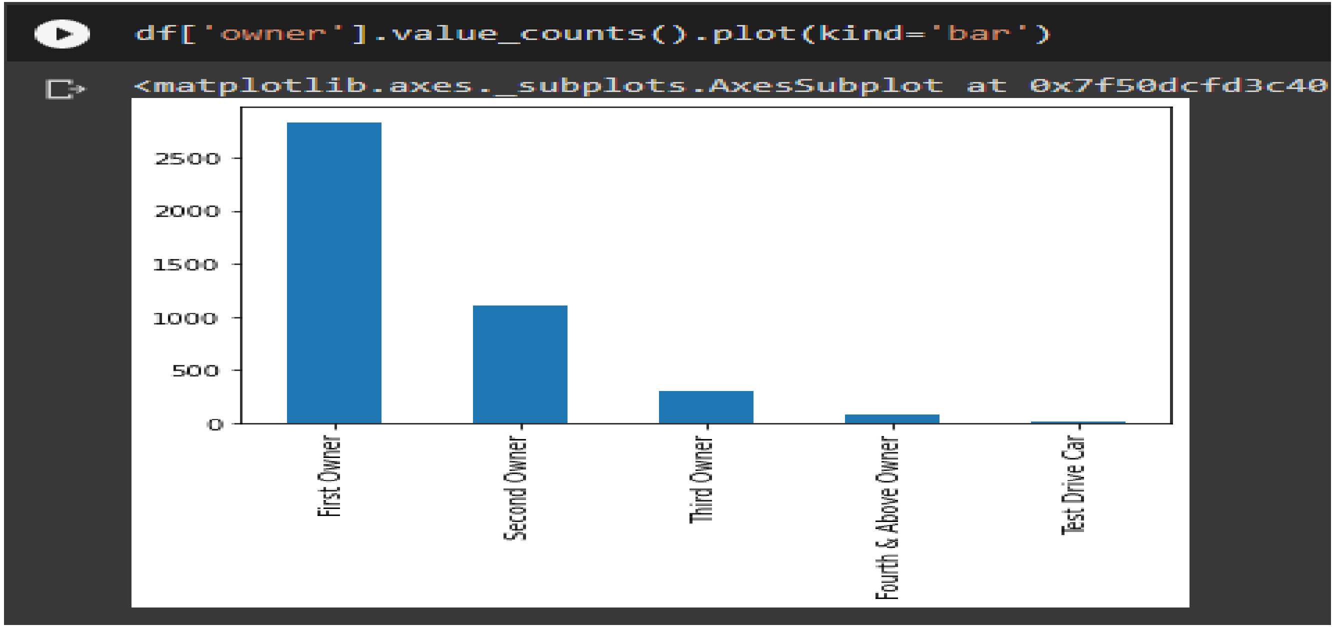


```
df['fuel'].value_counts().plot(kind='bar')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f50d
```

Dataset having more records of Diesel and Petrol.
Considering fuel columns we can say dataset is imbalanced.

Data insights



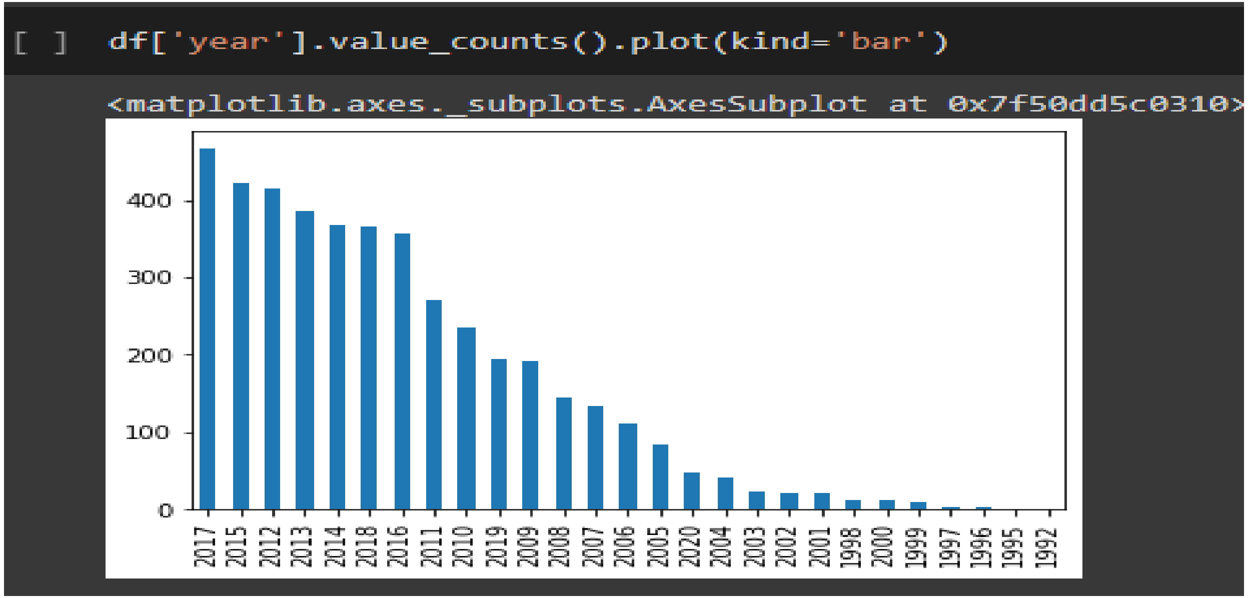
```
df['owner'].value_counts().plot(kind='bar')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f50dcfd3c40
```

Like fuel column owner column also have more records of First Owner

Data insights

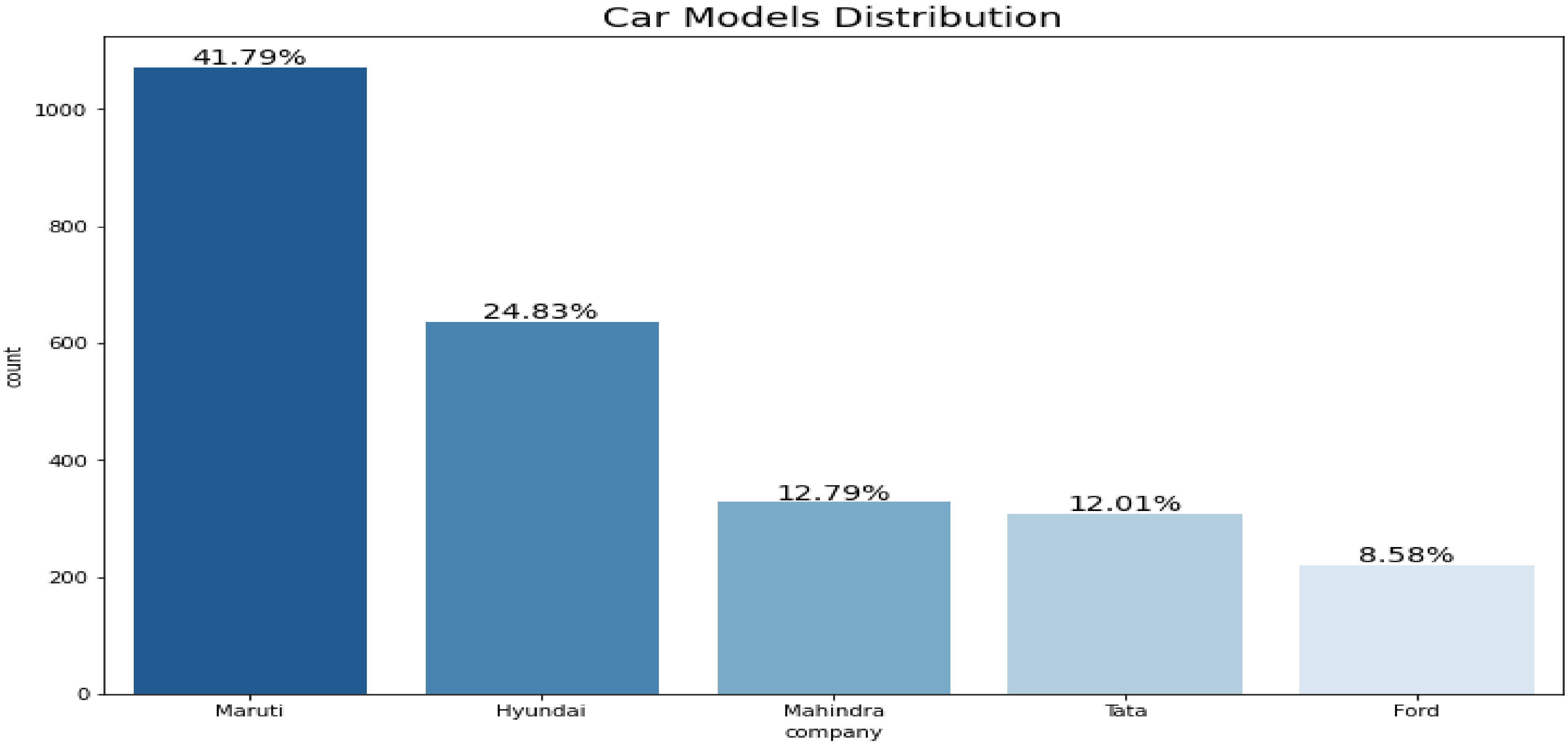
Year model distribution.



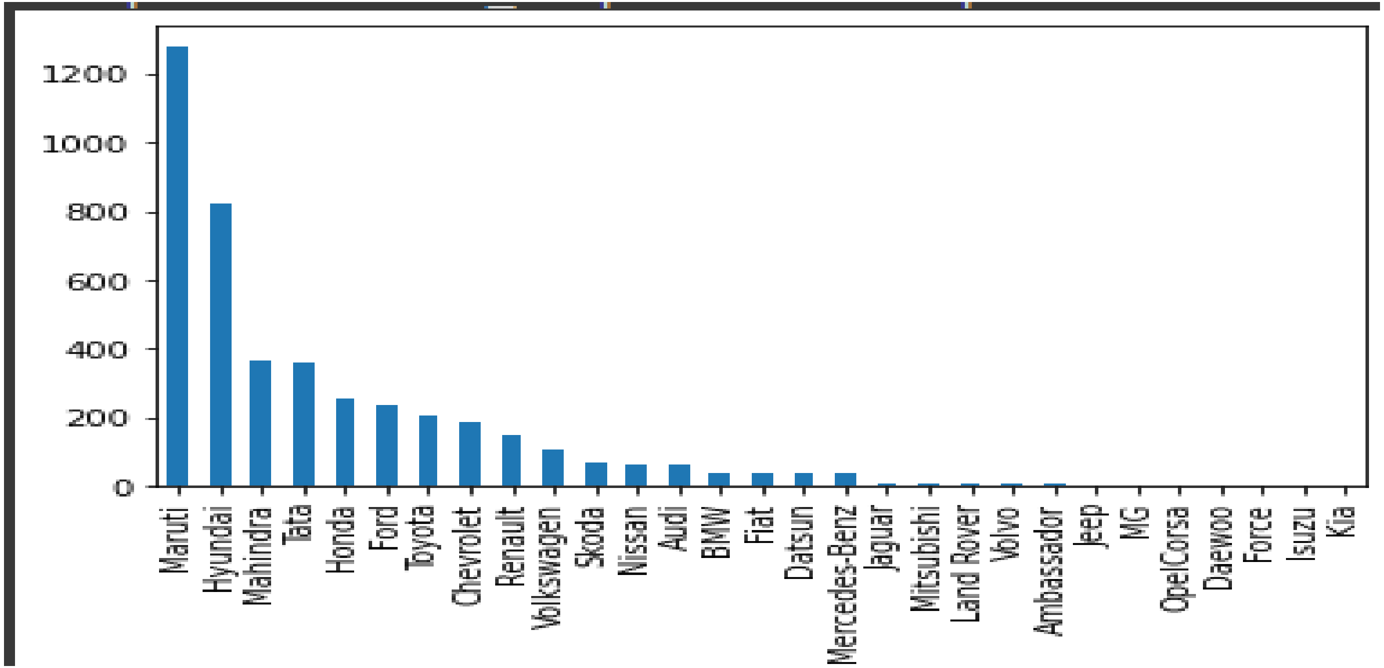
```
[ ]   df['year'].value_counts().plot(kind='bar')

      <matplotlib.axes._subplots.AxesSubplot at 0x7f50dd5c0310>
```

Data insights

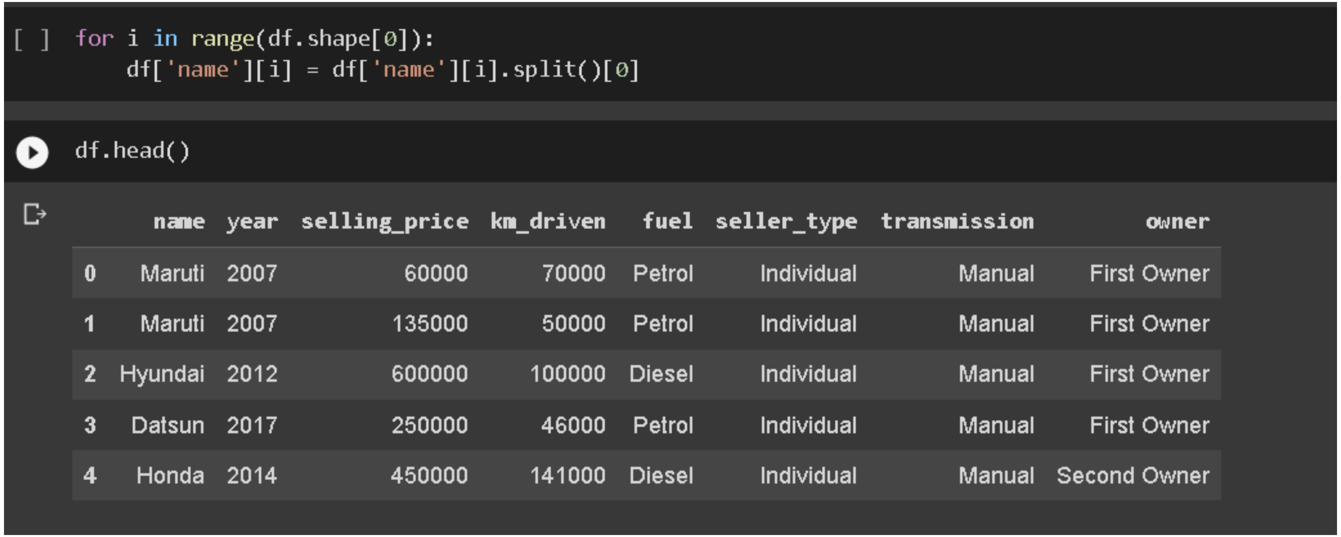


Data insights



Car model distribution.

Data Cleaning & Pre-Processing Steps.



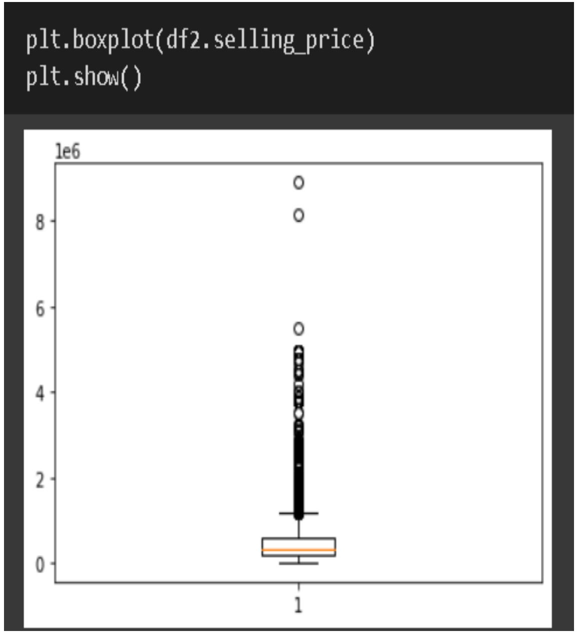
```python
for i in range(df.shape[0]):
    df['name'][i] = df['name'][i].split()[0]
```

```python
df.head()
```

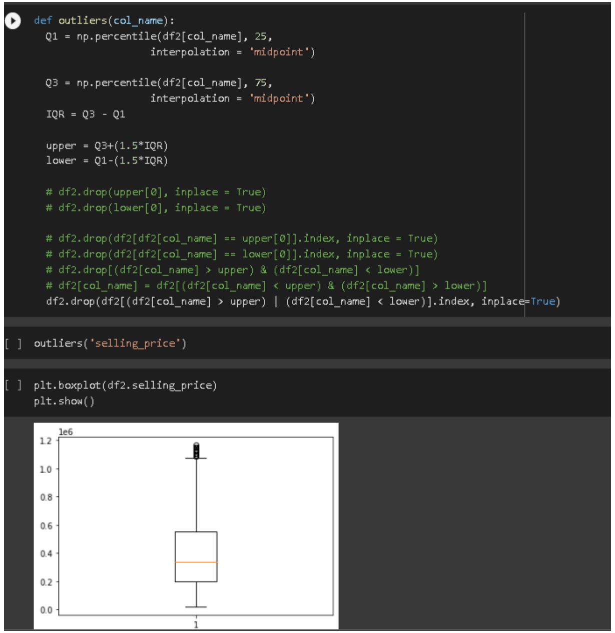
	name	year	selling_price	km_driven	fuel	seller_type	transmission	owner
0	Maruti	2007	60000	70000	Petrol	Individual	Manual	First Owner
1	Maruti	2007	135000	50000	Petrol	Individual	Manual	First Owner
2	Hyundai	2012	600000	100000	Diesel	Individual	Manual	First Owner
3	Datsun	2017	250000	46000	Petrol	Individual	Manual	First Owner
4	Honda	2014	450000	141000	Diesel	Individual	Manual	Second Owner

Extracted Manufacturer name from ‘name’ columns

Data Cleaning & Pre-Processing Steps.



```
plt.boxplot(df2.selling_price)
plt.show()
```



```
def outliers(col_name):
  Q1 = np.percentile(df2[col_name], 25,
                     interpolation = 'midpoint')

  Q3 = np.percentile(df2[col_name], 75,
                     interpolation = 'midpoint')
  IQR = Q3 - Q1

  upper = Q3+(1.5*IQR)
  lower = Q1-(1.5*IQR)

  # df2.drop(upper[0], inplace = True)
  # df2.drop(lower[0], inplace = True)

  # df2.drop(df2[df2[col_name] == upper[0]].index, inplace = True)
  # df2.drop(df2[df2[col_name] == lower[0]].index, inplace = True)
  # df2.drop[(df2[col_name] > upper) & (df2[col_name] < lower)]
  # df2[col_name] = df2[(df2[col_name] < upper) & (df2[col_name] > lower)]
  df2.drop(df2[(df2[col_name] > upper) | (df2[col_name] < lower)].index, inplace=True)
```

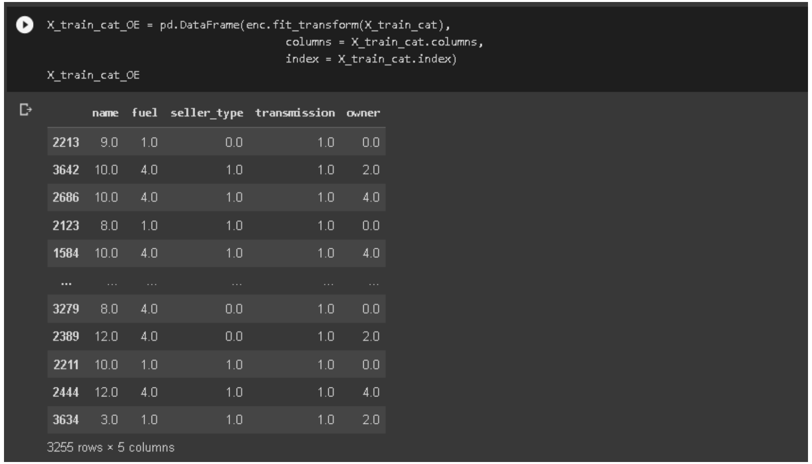
```
[ ] outliers('selling_price')
```

```
[ ] plt.boxplot(df2.selling_price)
    plt.show()
```

Outliers present in selling_price column

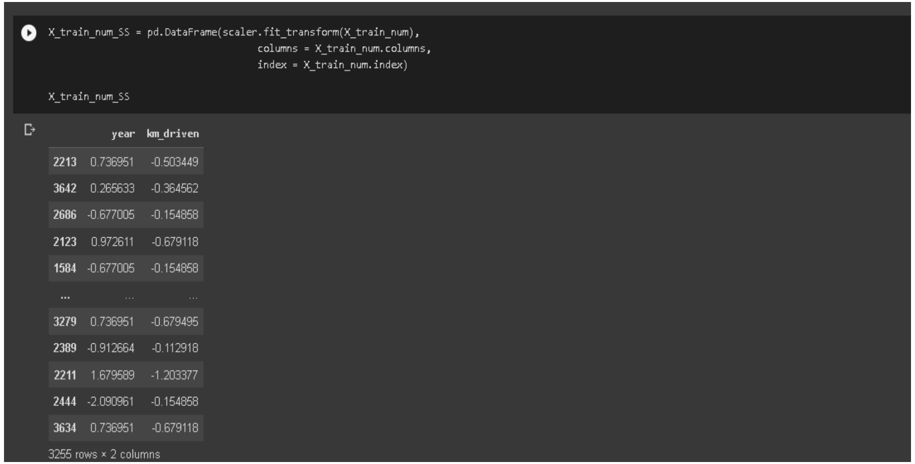
Treated Outliers

Data Cleaning & Pre-Processing Steps.



```
X_train_cat_OE = pd.DataFrame(enc.fit_transform(X_train_cat),
                             columns = X_train_cat.columns,
                             index = X_train_cat.index)
X_train_cat_OE
```

```
      name  fuel  seller_type  transmission  owner
2213   9.0   1.0          0.0           1.0    0.0
3642  10.0   4.0          1.0           1.0    2.0
2686  10.0   4.0          1.0           1.0    4.0
2123   8.0   1.0          1.0           1.0    0.0
1584  10.0   4.0          1.0           1.0    4.0
 ...   ...   ...          ...           ...    ...
3279   8.0   4.0          0.0           1.0    0.0
2389  12.0   4.0          0.0           1.0    2.0
2211  10.0   1.0          1.0           1.0    0.0
2444  12.0   4.0          1.0           1.0    4.0
3634   3.0   1.0          1.0           1.0    2.0
3255 rows × 5 columns
```



```
X_train_num_SS = pd.DataFrame(scaler.fit_transform(X_train_num),
                              columns = X_train_num.columns,
                              index = X_train_num.index)

X_train_num_SS
```

```
          year  km_driven
2213  0.736951  -0.503449
3642  0.265633  -0.364562
2686 -0.677005  -0.154858
2123  0.972611  -0.679118
1584 -0.677005  -0.154858
 ...       ...        ...
3279  0.736951  -0.679495
2389 -0.912664  -0.112918
2211  1.679589  -1.203377
2444 -2.090961  -0.154858
3634  0.736951  -0.679118
3255 rows × 2 columns
```

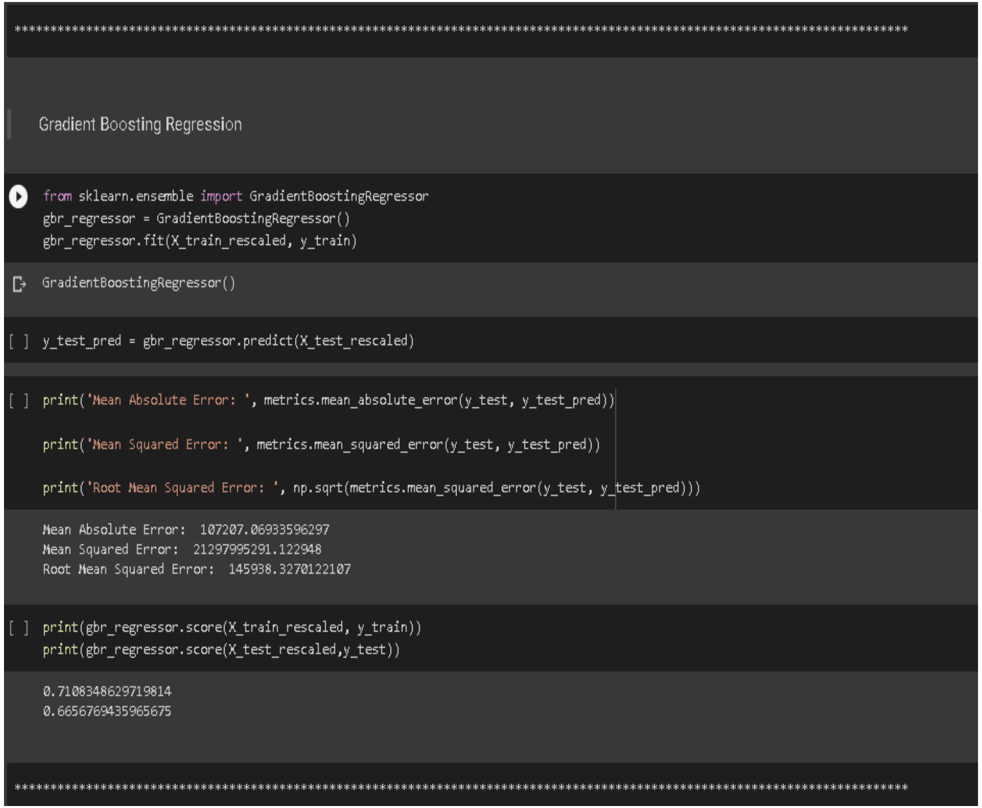
Firstly Splited data into train test,

After applied Ordinal encoding to categorical columns and Standard scalar to train and test data after spliting,

I had done scaling after the splitting process because if we do scaling before splitting, their can be changes of data lickeage and it can be affect over model.

ML Modeling

skill academy



```
*****************************************************************************************************************************
```

Gradient Boosting Regression

```
from sklearn.ensemble import GradientBoostingRegressor
gbr_regressor = GradientBoostingRegressor()
gbr_regressor.fit(X_train_rescaled, y_train)
```

```
GradientBoostingRegressor()
```

```
y_test_pred = gbr_regressor.predict(X_test_rescaled)
```

```
print('Mean Absolute Error: ', metrics.mean_absolute_error(y_test, y_test_pred))

print('Mean Squared Error: ', metrics.mean_squared_error(y_test, y_test_pred))

print('Root Mean Squared Error: ', np.sqrt(metrics.mean_squared_error(y_test, y_test_pred)))
```

```
Mean Absolute Error:  107207.06933596297
Mean Squared Error:  21297995291.122948
Root Mean Squared Error:  145938.3270122107
```

```
print(gbr_regressor.score(X_train_rescaled, y_train))
print(gbr_regressor.score(X_test_rescaled,y_test))
```

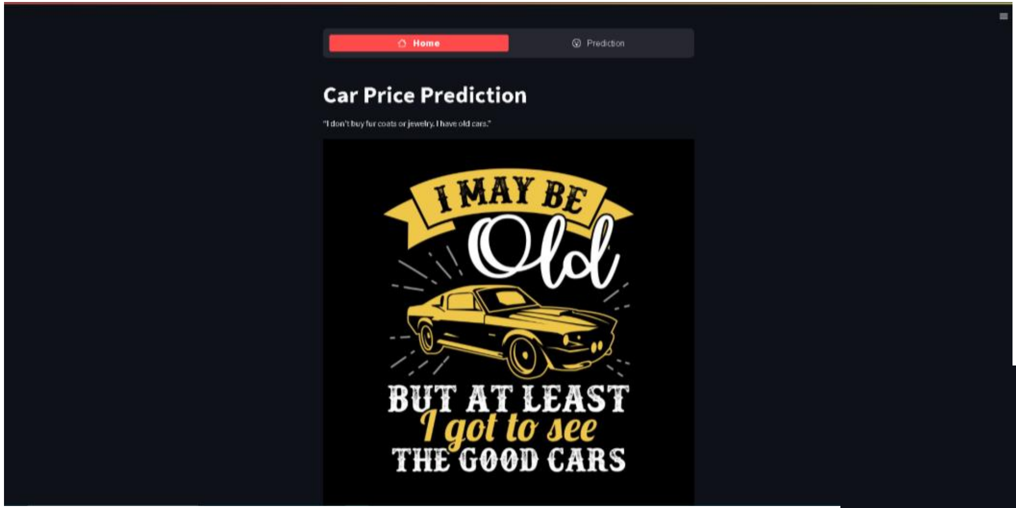
```
0.7108348629719814
0.6656769435965675
```

```
*****************************************************************************************************************************
```

I had trained various model like Random Forest, Decision Tree, Gradient Boosting, SVM, KNN Linear Regression, Lasso, Ridge.

Gradient Boosting is giving great accuracy on this dataset comparing other models, other Models are overfitting because I choose Gradient boosting model for this project based On train and test score.

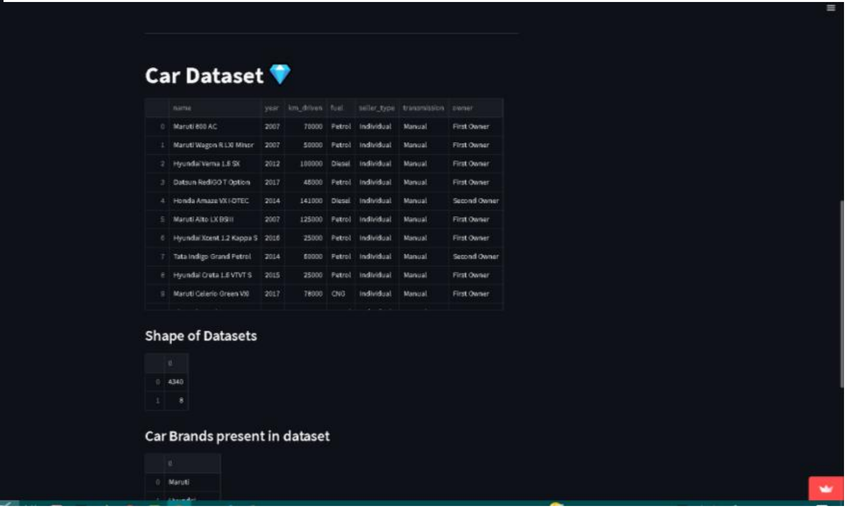
Deployment of ML Models using Streamlit.



Car Price Prediction

"I don't buy fur coats or jewelry. I have old cars."

Created web app for prediction using streamlit and deployed on same streamlit cloud platform



Car Dataset 💎

	name	year	km_driven	fuel	seller_type	transmission	owner
0	Maruti 800 AC	2007	70000	Petrol	Individual	Manual	First Owner
1	Maruti Wagon R LXI Minor	2007	50000	Petrol	Individual	Manual	First Owner
2	Hyundai Verna 1.6 SX	2012	100000	Diesel	Individual	Manual	First Owner
3	Datsun RediGO T Option	2017	46000	Petrol	Individual	Manual	First Owner
4	Honda Amaze VX i-DTEC	2014	141000	Diesel	Individual	Manual	Second Owner
5	Maruti Alto LX BSIII	2007	125000	Petrol	Individual	Manual	First Owner
6	Hyundai Xcent 1.2 Kappa S	2016	25000	Petrol	Individual	Manual	First Owner
7	Tata Indigo Grand Petrol	2014	60000	Petrol	Individual	Manual	Second Owner
8	Hyundai Creta 1.6 VTVT S	2015	25000	Petrol	Individual	Manual	First Owner
9	Maruti Celerio Green VXI	2017	78000	CNG	Individual	Manual	First Owner

Shape of Datasets

	0
0	4340
1	8

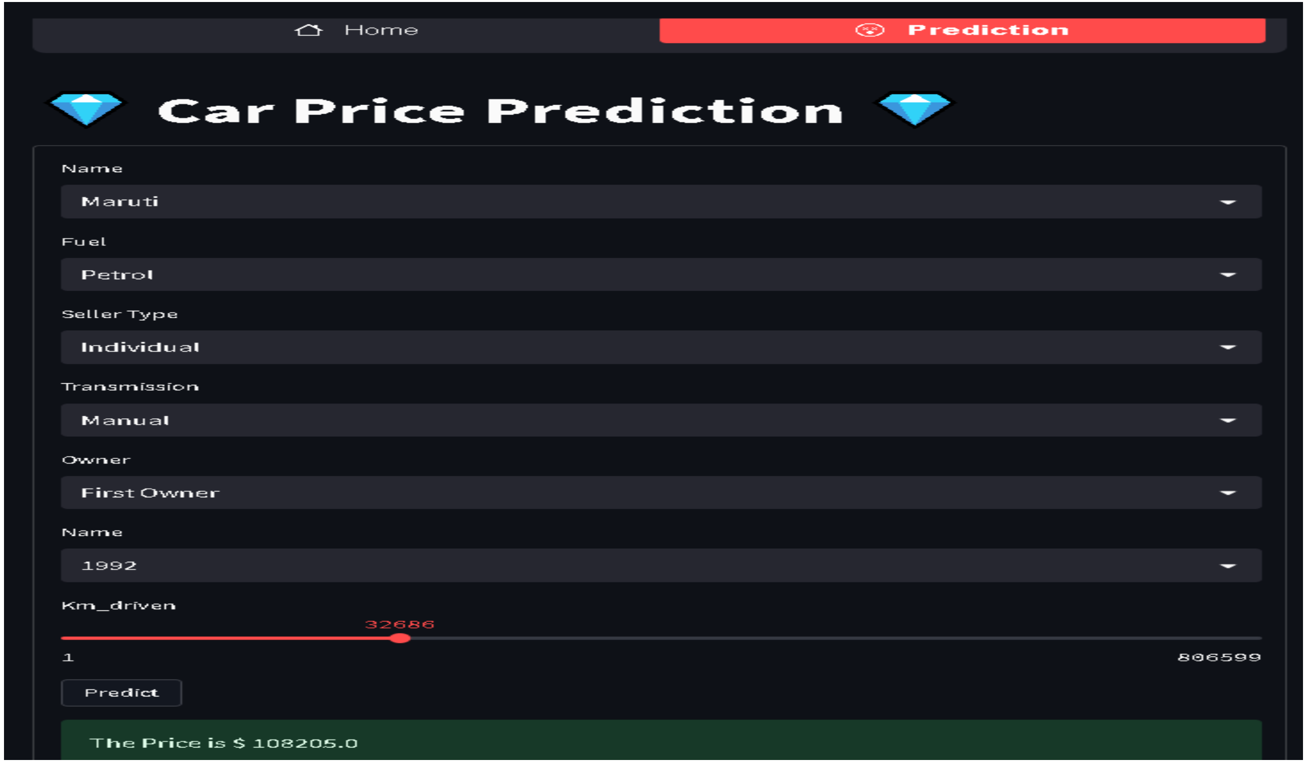
Car Brands present in dataset

	0
0	Maruti
1	Hyundai

This is first page of webapp containing basic Information of dataset.

Deployment of ML Models using Streamlit.

skill academy



💎 Car Price Prediction 💎

Name: Maruti

Fuel: Petrol

Seller Type: Individual

Transmission: Manual

Owner: First Owner

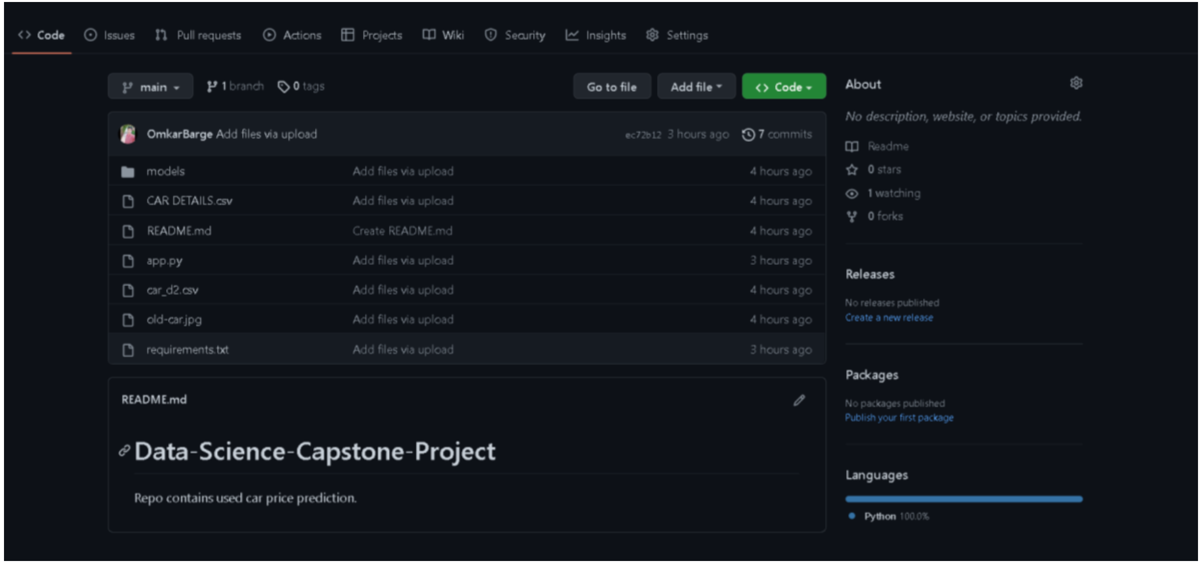
Name: 1992

Km_driven: 32686 (1 – 806599)

The Price is $ 108205.0

This is main page of our web app which is use for predictions.

Github



OmkarBarge Add files via upload ec72b12 3 hours ago 7 commits

models	Add files via upload	4 hours ago
CAR DETAILS.csv	Add files via upload	4 hours ago
README.md	Create README.md	4 hours ago
app.py	Add files via upload	3 hours ago
car_d2.csv	Add files via upload	4 hours ago
old-car.jpg	Add files via upload	4 hours ago
requirements.txt	Add files via upload	3 hours ago

README.md

Data-Science-Capstone-Project

Repo contains used car price prediction.

About

No description, website, or topics provided.

Readme

0 stars

1 watching

0 forks

Releases

No releases published

Create a new release

Packages

No packages published

Publish your first package

Languages

Python 100.0%

Uploaded all files regarding this project to github along with best model.

link : - https://github.com/OmkarBarge/Data-Science-Capstone-Project

Endnotes

Reference Links:-

Githib link: https://github.com/OmkarBarge/Data-Science-Capstone-Project
Streamlit: https://used-cars-price-prediction.streamlit.app/

Endnotes

Thank You

Omkar S Barge