Team Members

Name	ID
اسامه سمير عبدالمنعم	20210140
مروه عمر محمد محمود	20210900
رنا عصام الدين عيسى	20210335
احمد محمد عبدالسلام محمد	20210102
عبدالله محمد سعيد	20210566
عمر ناصر جمال	20210628
تقى محمد احمد	20210242

Car details dataset

Link in Kaggle:

Vehicle Dataset | Kaggle

Dataset Description:

it contains about 8129 car with features:

[name,year,selling_price,km_driven,fuel,seller_type, transmition,owner,mileage,engine,max_power,torque, seats]

I have found about 215 to 222 null values from 8129,

So I have filled them using enable_iterative_imputer class that found in sklearn.experimental
also in wrangling data I have found that some features contain symbols and measuring unit so it makes the columns as categorical so I removed all of them like that:

mileage	engine	max_power	torque
23.4 kmpl	1248 CC	74 bhp	190Nm@ 2000rpm
21.14 kmpl	1498 CC	103.52 bhp	250Nm@ 1500-2500rpm
17.7 kmpl	1497 CC	78 bhp	12.7@ 2,700(kgm@ rpm)
23.0 kmpl	1396 CC	90 bhp	22.4 kgm at 1750-2750rpm
16.1 kmpl	1298 CC	88.2 bhp	11.5@ 4,500(kgm@ rpm)

Python

km_driven	seats	mileage	engine	max_power	torque
145,500.00	5.00	23.40	1,248.00	74.00	2,000.00
120,000.00	5.00	21.14	1,498.00	103.52	2,500.00
140,000.00	5.00	17.70	1,497.00	78.00	2,700.00
127,000.00	5.00	23.00	1,396.00	90.00	2,750.00
120,000.00	5.00	16.10	1,298.00	88.20	4,500.00

So I could use them as a numerical features.

Also in feature [name] I split it into three features

To get more features which may be used in the models

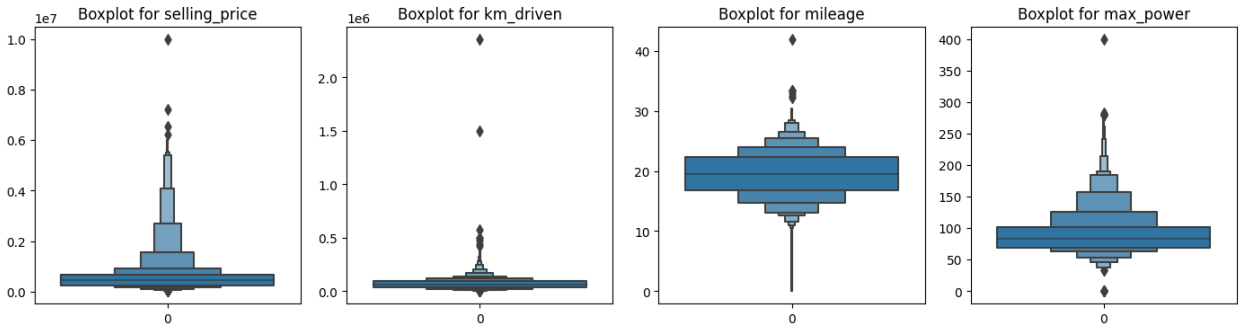
Before:

	name
	Maruti Swift Dzire VDI
	Skoda Rapid 1.5 TDI Ambition
	Honda City 2017-2020 EXi
	Hyundai i20 Sportz Diesel
	Maruti Swift VXI BSIII

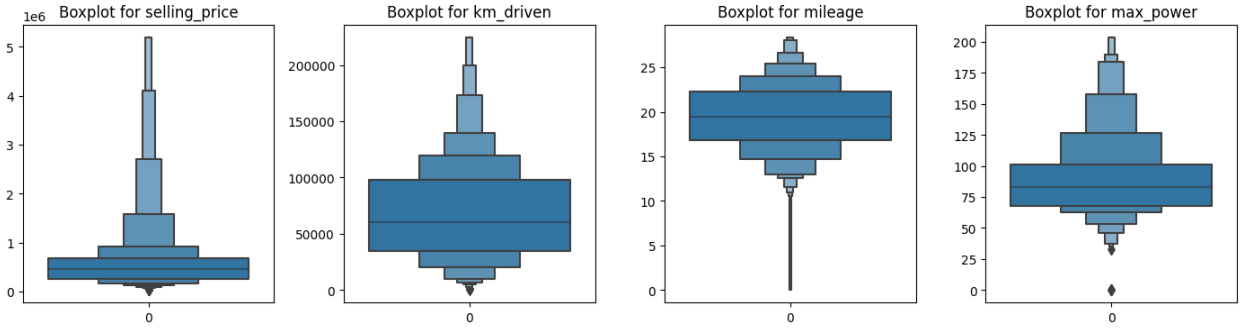
After:

	brand	model	variant	
2164	Maruti	Swift	VXI 2018	P
7740	Hyundai	Santro-Xing	XG	P
3499	Renault	KWID	RXL	P
7309	Maruti	Ertiga	VDI	D
6658	Mahindra	TUV-300	T8	D

Then I detected if there outliers using boxenplots so
there the shape of some of columns before deleting outliers:



And this the shape after deleting the outliers



After this I encoded the categorical features into numerical features and I have a multiple nonunique names so I use Target Encoding which is suitable for my data values

Data before encoding:

brand	model	variant	fuel	seller_type	transmission	owner	year	selling_price	km_driven	seats	mileage	engine	max_power	torque
Maruti	Swift	VXI 2018	Petrol	Individual	Manual	Second Owner	2018	509999	70000	5	22.00	1197	81.80	4200
Hyundai	Santro-Xing	XG	Petrol	Individual	Manual	Second Owner	2004	70000	70000	5	19.42	1457	91.51	3082
Renault	KWID	RXL	Petrol	Individual	Manual	First Owner	2018	350000	40000	5	25.17	799	53.30	4386
Maruti	Ertiga	VDI	Diesel	Individual	Manual	First Owner	2015	650000	70000	7	20.77	1248	88.76	1750
Mahindra	TUV-300	T8	Diesel	Individual	Manual	First Owner	2015	500000	120000	7	18.49	1493	100.00	2800

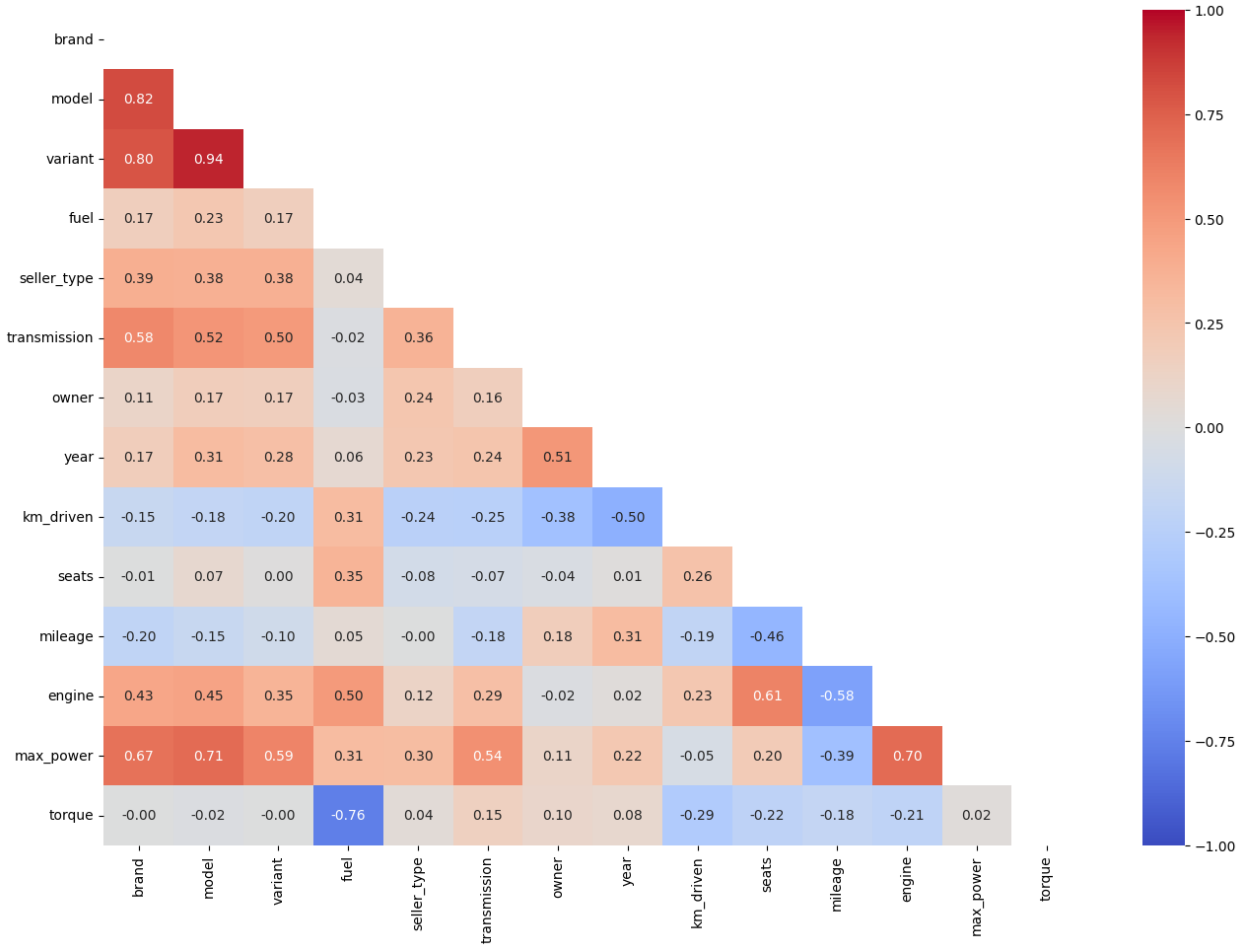
After encoding:

	brand	model	variant	fuel	seller_type	transmission	owner	year	selling_price	km_driven	seats	mileage	engine	max_power	torque
0	403,075.72	512,518.47	536,322.13	785,589.68	497,236.96	455,425.86	777,298.93	2014	450000	145500	5	23.40	1248	74.00	2000
1	607,729.14	596,754.45	619,588.69	785,589.68	497,236.96	455,425.86	392,964.47	2014	370000	120000	5	21.14	1498	103.52	2500
2	596,178.01	570,062.26	533,687.97	460,273.89	497,236.96	455,425.86	284,015.33	2006	158000	140000	5	17.70	1497	78.00	2700
3	458,554.37	520,630.47	580,897.18	785,589.68	497,236.96	455,425.86	777,298.93	2010	225000	127000	5	23.00	1396	90.00	2750
4	403,075.72	456,304.40	525,459.35	460,273.89	497,236.96	455,425.86	777,298.93	2007	130000	120000	5	16.10	1298	88.20	4500
5	458,554.37	493,431.78	610,846.47	460,273.89	497,236.96	455,425.86	777,298.93	2017	440000	45000	5	20.14	1197	81.86	4000
6	403,075.72	261,877.09	536,019.04	261,942.88	497,236.96	455,425.86	777,298.93	2007	96000	175000	5	17.30	1061	57.50	4500
7	403,075.72	68,798.03	557,477.65	460,273.89	497,236.96	455,425.86	392,964.47	2001	45000	5000	4	16.10	796	37.00	2500
8	959,946.67	488,720.23	593,329.54	785,589.68	497,236.96	455,425.86	777,298.93	2011	350000	90000	5	23.59	1364	67.10	2400
9	516,682.58	308,928.45	532,299.36	785,589.68	497,236.96	455,425.86	777,298.93	2013	200000	169000	5	20.00	1399	68.10	2000

Then I have scaled the data using StandardScaler

	brand	model	variant	fuel	seller_type	transmission	owner	year	km_driven	seats	mileage	engine	max_power	torque
0	-0.35	-0.14	-0.20	0.92	-0.43	-0.39	0.72	0.05	1.72	-0.43	1.01	-0.42	-0.50	-1.22
1	-0.03	-0.00	-0.05	0.92	-0.43	-0.39	-1.19	0.05	1.15	-0.43	0.44	0.08	0.36	-0.66
2	-0.05	-0.05	-0.21	-1.04	-0.43	-0.39	-1.73	-1.93	1.59	-0.43	-0.43	0.08	-0.39	-0.44
3	-0.27	-0.13	-0.12	0.92	-0.43	-0.39	0.72	-0.94	1.30	-0.43	0.91	-0.12	-0.03	-0.38
4	-0.35	-0.23	-0.22	-1.04	-0.43	-0.39	0.72	-1.68	1.15	-0.43	-0.83	-0.32	-0.09	1.58

correlations between the features



during this heatmap target variable[selling_price]

```
 is correlated with [model], [variant] and [max_power] features. So, I drop them for model implementation.
```

Then our used features are:[brand,seller_type,fuel,transmition,owner,year,km_driven,seats,mileage,engine,torque]

And there shape are: (8129,11)

While our target is[selling_price]

And its shape is: (8129,1)

Then splitting the data level:

I have used cross validation (k folds) with k=5

The assign the variables with train_test_split

Function,I give it parameters:test_size=50%

Because I noticed that increasing the score of testing while training remains constant at increasing or decreasing the size
and in target variable (y) I assigned to it [selling_price] column which my target to make the model predict these values
and in in features(x) I assigned to it all remaining columns except [selling_price].

Then I scaled the data after splitting at:

X_train,x_val,x_test and

Y_train,y_val,y_test

The the data is ready to be in the models.

a) Linear Regression

b) Goal : predict car prices

c) Code : used linear regression from sklearn.lineae_model

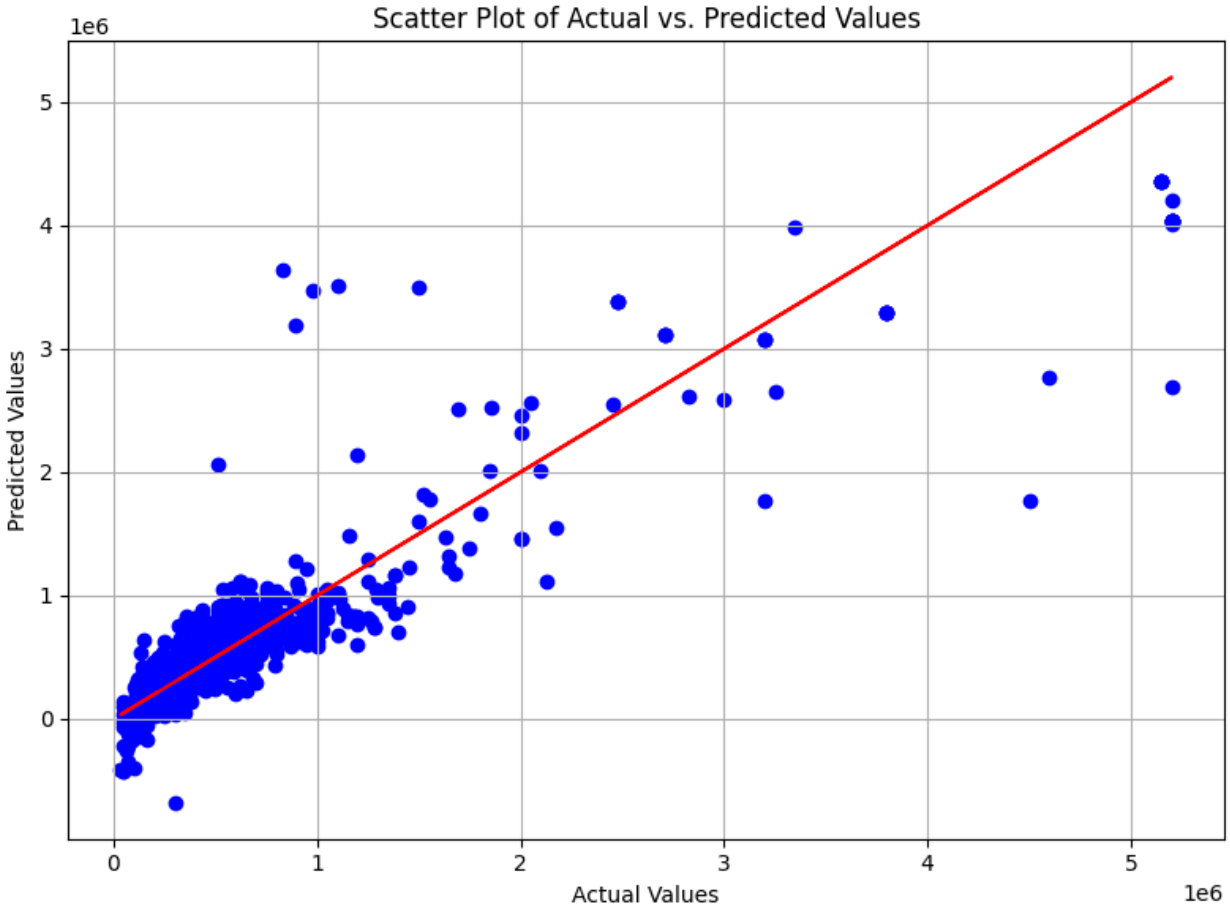
d) Cross validation : "k-fold cross-validation" with k=5

e) Result :

```
Training score: 0.8248603949504367,
Testing score: 0.8173415139261601

Mean Absolute Error (MAE): 196795.9227
Mean Squared Error (MSE): 124595019507.0392
Root Mean Squared Error (RMSE): 352980.1970
R-squared (R²) score: 0.8157
```

And this the plotting of the model:



K Nearest Neighbors (KNN)

Goal : predict car prices

Selecting the best K and parameters using gridsearch I get these results

```
Best Parameters: {'algorithm': 'brute', 'metric':
'manhattan', 'n_neighbors': 10, 'weights':
'distance'}
Best Score: 0.9523781552866419
Best Estimator:
KNeighborsRegressor(algorithm='brute',
metric='manhattan', n_neighbors=10,
                    weights='distance')
```

Cross validation : "k-fold cross-validation" with k=5

Result :

Training score: 0.9995729563681929
,Testing score: 0.9356555596021242

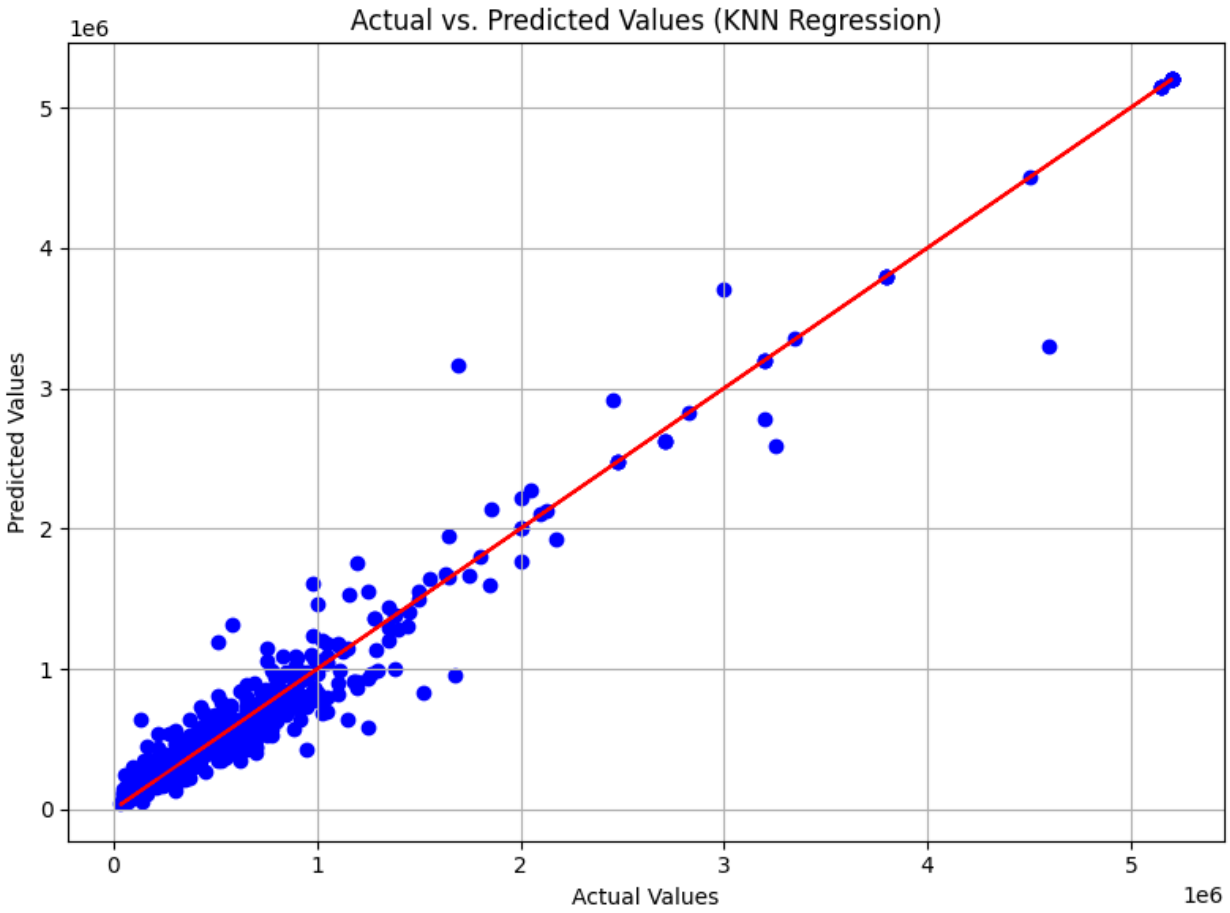
Mean Absolute Error (MAE): 79059.4671

Mean Squared Error (MSE): 21313920629.9886

Root Mean Squared Error (RMSE): 145992.8787

R-squared (R^2) score: 0.9685

And this the plotting of the model:



According as using Metrics,score,plotting we get that using Knn is better than Linear Regression here.