Car Price Prediction

Submitted by:

Himani Uniyal

ACKNOWLEDGMENT

I am really grateful for this project opportunity and would sincerely thank to “Flip Robo” for trusting me with this project.

Besides this, I would like to express my gratitude towards Srishti Mann Ma’am (SME) to provide us quick revert on the queries raised and continuous support throughout the project.

.

INTRODUCTION

- **Business Problem Framing**
  With the change in market due to Covid 19 impact, Car market is facing problems with their previous car price valuation machine learning models. Here we will develop new machine learning models from new data to evaluate car price.
- **Conceptual Background of the Domain Problem**
  Generally when cars are in demand we making them costly and some are not in demand hence cheaper. So on the basis of data we need to analyse how the market work in case of used cars.
- **Review of Literature**
  The data study shows that the big cities having high living index, have costlier cars. Also the older the car is lesser is the valuation. The number of owners the car has does not have much impact in the prediction of price.
- **Motivation for the Problem Undertaken**
  It is a new challenge to scrap the data, and then develop the model on the basis of data.

Analytical Problem Framing

- **Mathematical/ Analytical Modeling of the Problem**

The data set shape is 251*9 columns.

```
1 df1
```

	Car	History	KM	Year_purchase	Owner	fuel	Transmission	Price	Location
1	2018 Volkswagen Ameo HIGHLINE PLUS 1.0 MANUAL	Non-Accidental	65235	3	1st Owner	Petrol	MANUAL	3.34	hyderabad
2	2017 Skoda Octavia Style 1.4 TSI MT MANUAL	Non-Accidental	47264	4	1st Owner	Petrol	MANUAL	6.22	hyderabad
3	2016 Maruti Baleno ZETA 1.2 K12 MANUAL	Non-Accidental	103354	5	1st Owner	Petrol	MANUAL	13.05	hyderabad
4	2015 Honda Jazz 1.2 V AT AUTOMATIC	Non-Accidental	65571	6	2nd Owner	Petrol	AUTOMATIC	6.22	hyderabad
5	2017 Maruti Baleno DELTA 1.2 K12 MANUAL	Non-Accidental	22876	4	1st Owner	Petrol	MANUAL	5.24	hyderabad
...	...	...	...	...	...	...	...	...	...
257	2016 Maruti Ertiga ZDI SHVS MANUAL	Non-Accidental	66982	5	2nd Owner	Diesel	MANUAL	8.17	Mumbai
258	2017 Hyundai i20 Active 1.2 SX MANUAL	Non-Accidental	12910	4	1st Owner	Petrol	MANUAL	6.49	Mumbai
259	2017 Maruti Vitara Brezza VDI OPT MANUAL	Non-Accidental	46304	4	1st Owner	Diesel	MANUAL	7.45	Mumbai
260	2015 Maruti Ciaz ZXI PLUS MANUAL	Non-Accidental	23382	6	1st Owner	Petrol	MANUAL	6.55	Mumbai
261	2017 Jeep Compass 2.0 LONGITUDE MANUAL	Non-Accidental	62695	4	1st Owner	Diesel	MANUAL	13.78	Mumbai

251 rows × 9 columns

Description of data set:

```
1 df_encoded.describe()
```

	Car	Year_purchase	Owner	fuel	Transmission	Price	Location
count	251.000000	251.000000	251.000000	251.000000	251.000000	251.000000	251.000000
mean	87.856574	5.139442	0.247012	0.541833	0.868526	6.182510	0.984064
std	51.584526	2.361457	0.484501	0.499242	0.338593	3.062169	0.963195
min	0.000000	0.000000	0.000000	0.000000	0.000000	2.370000	0.000000
25%	45.000000	3.000000	0.000000	0.000000	1.000000	4.110000	0.000000
50%	88.000000	5.000000	0.000000	1.000000	1.000000	5.270000	1.000000
75%	133.500000	7.000000	0.000000	1.000000	1.000000	7.450000	2.000000
max	176.000000	11.000000	2.000000	1.000000	1.000000	19.260000	2.000000

The problem is a regression problem and we have to predict the price of the data.

- **Data Sources and their formats**

  Data Set:

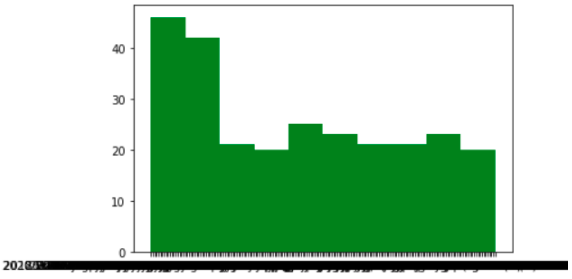
```
df1
```

	Car	History	KM	Year_purchase	Owner	fuel	Transmission	Price	Location
1	2018 Volkswagen Ameo HIGHLINE PLUS 1.0 MANUAL	Non-Accidental	65235	3	1st Owner	Petrol	MANUAL	3.34	hyderabad
2	2017 Skoda Octavia Style 1.4 TSI MT MANUAL	Non-Accidental	47264	4	1st Owner	Petrol	MANUAL	6.22	hyderabad
3	2016 Maruti Baleno ZETA 1.2 K12 MANUAL	Non-Accidental	103354	5	1st Owner	Petrol	MANUAL	13.05	hyderabad
4	2015 Honda Jazz 1.2 V AT AUTOMATIC	Non-Accidental	65571	6	2nd Owner	Petrol	AUTOMATIC	6.22	hyderabad
5	2017 Maruti Baleno DELTA 1.2 K12 MANUAL	Non-Accidental	22876	4	1st Owner	Petrol	MANUAL	5.24	hyderabad
...	...	...	...	...	...	...	...	...	...
257	2016 Maruti Ertiga ZDI SHVS MANUAL	Non-Accidental	66982	5	2nd Owner	Diesel	MANUAL	8.17	Mumbai
258	2017 Hyundai i20 Active 1.2 SX MANUAL	Non-Accidental	12910	4	1st Owner	Petrol	MANUAL	6.49	Mumbai
259	2017 Maruti Vitara Brezza VDI OPT MANUAL	Non-Accidental	46304	4	1st Owner	Diesel	MANUAL	7.45	Mumbai
260	2015 Maruti Ciaz ZXI PLUS MANUAL	Non-Accidental	23382	6	1st Owner	Petrol	MANUAL	6.55	Mumbai
261	2017 Jeep Compass 2.0 LONGITUDE MANUAL	Non-Accidental	62695	4	1st Owner	Diesel	MANUAL	13.78	Mumbai

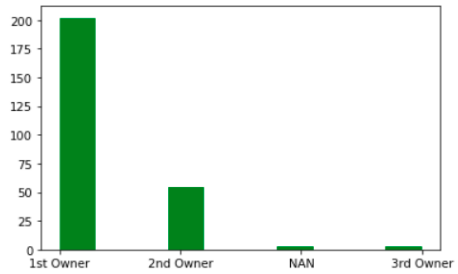
251 rows × 9 columns

The column History has only one type of data and will not make much influence in the data set, so it is dropped.

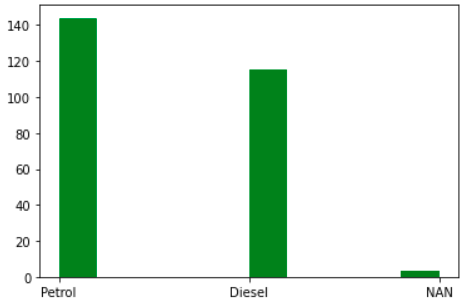
```
plt.hist(car['Car'],color='g')
figsize=(30,30)
plt.show()
```



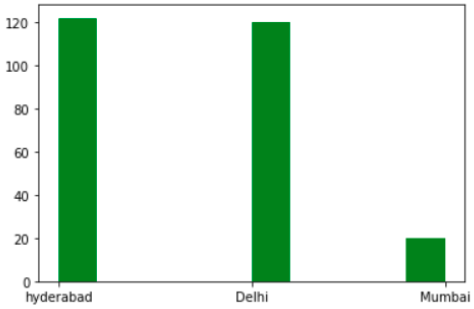

```
plt.hist(car['Owner'],color='g')
figsize=(30,30)
plt.show()
```




```
plt.hist(car['fuel'],color='g')
figsize=(30,30)
plt.show()
```

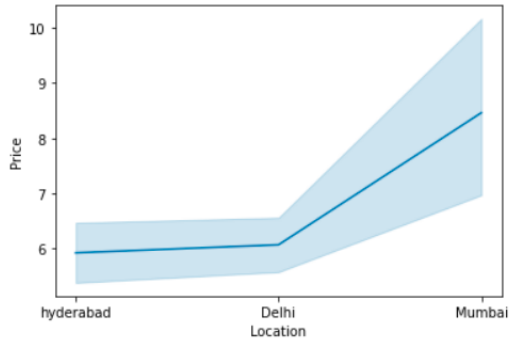



```
plt.hist(car['Location'],color='g')
figsize=(30,30)
plt.show()
```

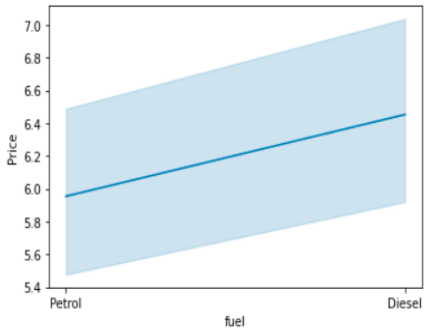


Further the visualization of each column has been done. Also each attribute is visualized with the price.

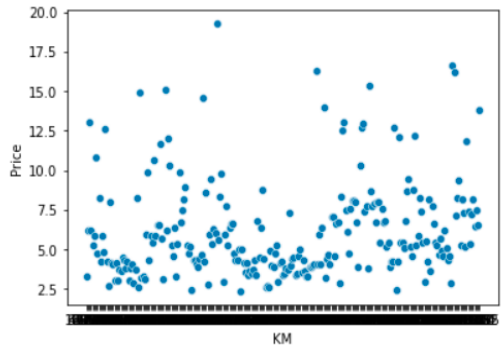
```
sns.lineplot(x="Location", y="Price",data=df2)
plt.show()
```



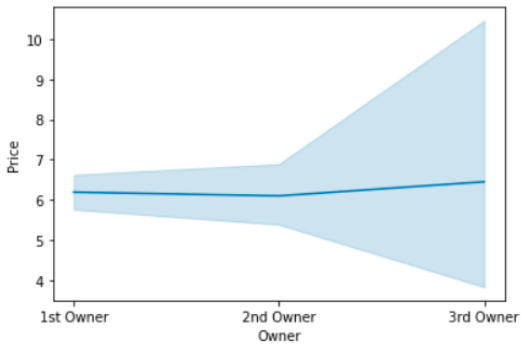

```
sns.lineplot(x="fuel", y="Price",data=df2)
plt.show()
```



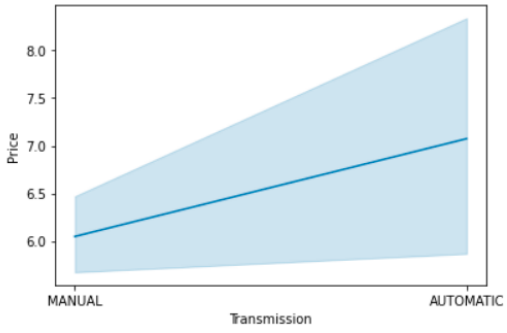

```
sns.scatterplot(x="KM", y="Price",data=df2)
plt.show()
```



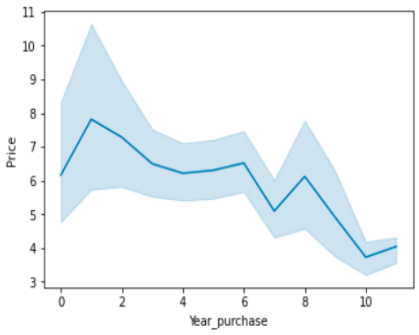

```
sns.lineplot(x="Owner", y="Price",data=df2)
plt.show()
```



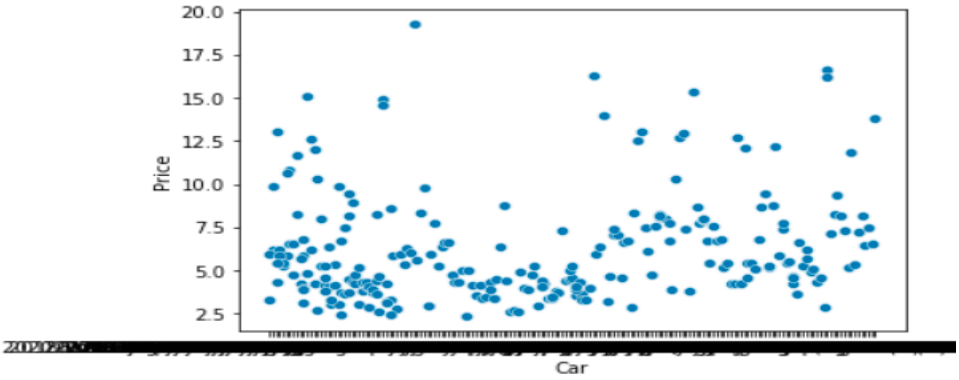

```
sns.lineplot(x="Transmission", y="Price",data=df2)
plt.show()
```




```
sns.lineplot(x="Year_purchase", y="Price",data=df2)
plt.show()
```




```
sns.scatterplot(x="Car", y="Price",data=df2)
plt.show()
```



- **Data Preprocessing Done**

  The data cleaning was executed,

```
#Splitted by space and removed kms
data['KM']=data['KM'].str.split(' ',expand=True)[0]
```

```
car1=data.assign(Price=data['Price'].str.replace(r'₹', ''))
```

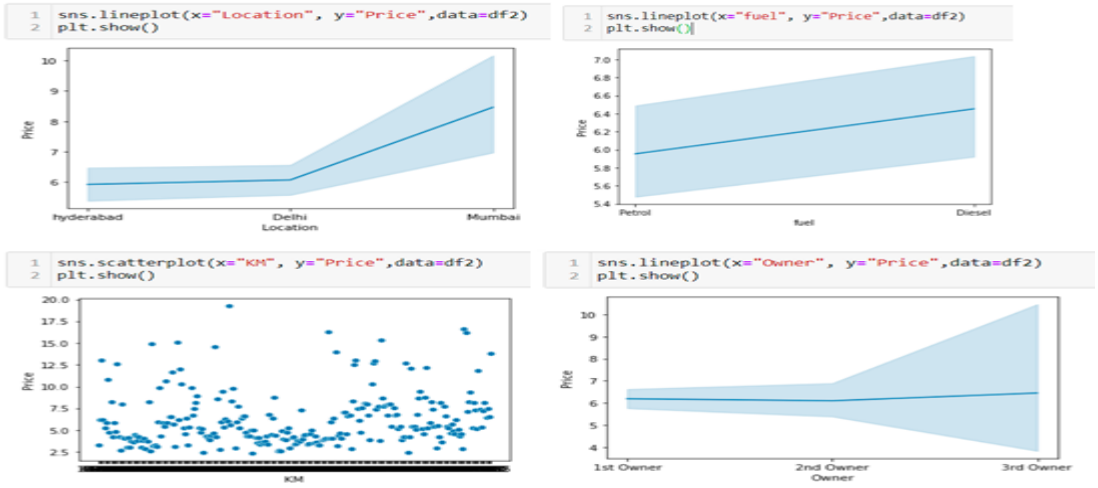
  The NAN values are replaced, the unwanted columns were dropped. And the data was made ready for further evaluation.

```
df_encoded.head()
```

	Car	KM	Year_purchase	Owner	fuel	Transmission	Price	Location
1	152	65235	3	0	1	1	3.34	2
2	130	47264	4	0	1	1	6.22	2
3	88	103354	5	0	1	1	13.05	2
4	54	65571	6	1	1	0	6.22	2
5	118	22876	4	0	1	1	5.24	2

- **Data Inputs- Logic- Output Relationships**

  Most of the attributes shows direct relation with the price.



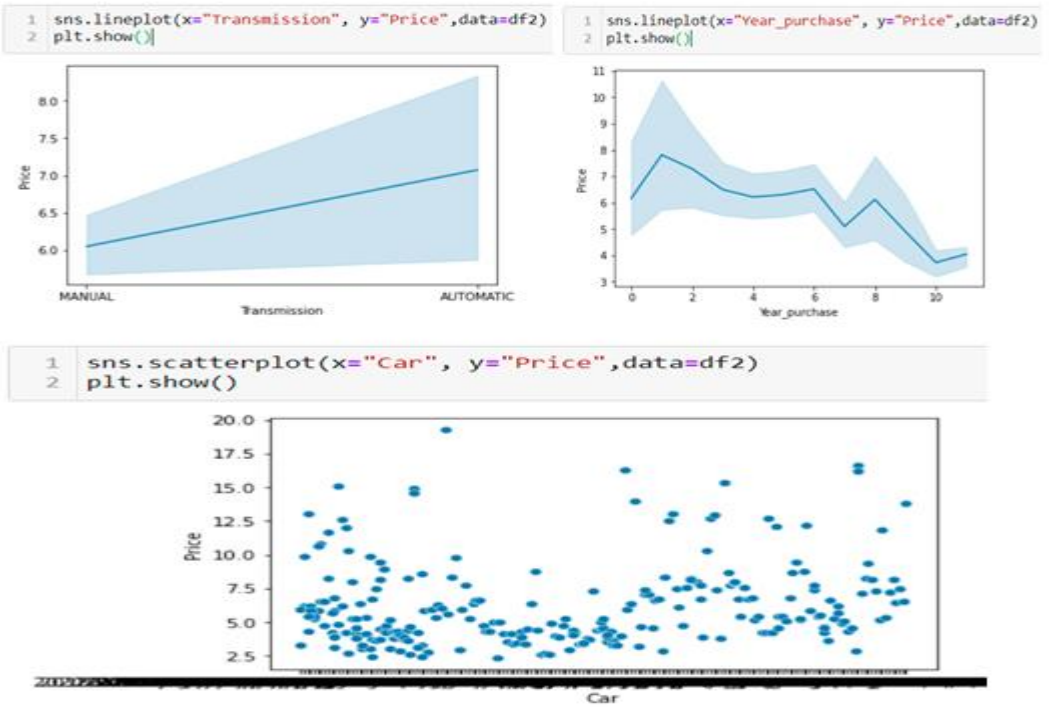
  Te following observations were made:

1. The vehicles in the Mumbai as more resale values.

2. Diesel vehicle has more expensive.

3. Distance covered did not give a better picture and prices are spread, and majorly constant.

4. The number of owner doesnot make much cange in the price of the vehicle.



5. Further the automatic vehicles are more in expensive, while the older the car is lesser is the price.

6. Also the car models are uniformly distributed, with some exceptions.

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

  Due to scrapping problem the data set is small to predict the better model.

- **Hardware and Software Requirements and Tools Used**
- Laptop, router, Python, Machine learning algorithms. Following libraries:
- import seaborn as sns
- from matplotlib import pyplot as plt
- import pandas as pd

- from sklearn.model_selection import train_test_split
- from sklearn.metrics import r2_score,mean_squared_error, mean_absolute_error
- from sklearn.linear_model import LinearRegression
- from sklearn.svm import SVR
- from sklearn.ensemble import RandomForestRegressor,GradientBoostingRegressor.
- from sklearn.model_selection import cross_val_score
- from sklearn.model_selection import GridSearchCV

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

  Firstly it was analysed that the data is a regression problem. After that the data preprocessing done and data is analysed and required algorithms were applied.

- Testing of Identified Approaches (Algorithms)
  The following models were used
- RandomForestRegressor
- GradientBoostingRegressor.
- LinearRegression

- Run and Evaluate selected models

```
from sklearn.model_selection import cross_val_score
ml_models=[LinearRegression(),SVR(),RandomForestRegressor(),GradientBoostingRegressor()]
for m in ml_models:
    m.fit(x_train,y_train)
    predm=m.predict(x_test)
    mse=mean_squared_error(y_test,predm)
    mae=mean_absolute_error(y_test,predm)
    r2=r2_score(y_test,predm)
    print(f'metrics of {m}:')
    print(f' mean_absolute_error: {mae}\n mean_squared_error: {mse}\n r2_score: {r2} ')
    score=cross_val_score(m,x_scaled,y, cv=5)
    print(' mean cv score:',score.mean())
    print('\n\n')
```

```
metrics of LinearRegression():
 mean_absolute_error: 2.6119050862281252
 mean_squared_error: 14.394900456087134
 r2_score: -0.03032565299391332
 mean cv score: -0.3301251089962346


metrics of SVR():
 mean_absolute_error: 2.3539192326042593
 mean_squared_error: 15.204625144065323
 r2_score: -0.08828229676727362
 mean cv score: -0.3415959586445788


metrics of RandomForestRegressor():
 mean_absolute_error: 2.598465079365079
 mean_squared_error: 15.193673784920636
 r2_score: -0.08749844513201266
 mean cv score: -0.39352948908315893


metrics of GradientBoostingRegressor():
 mean_absolute_error: 2.7522722970940428
 mean_squared_error: 15.244387968556072
 r2_score: -0.0911283503564102
 mean cv score: -0.4528786721739417
```

Key Metrics for success in solving problem under consideration

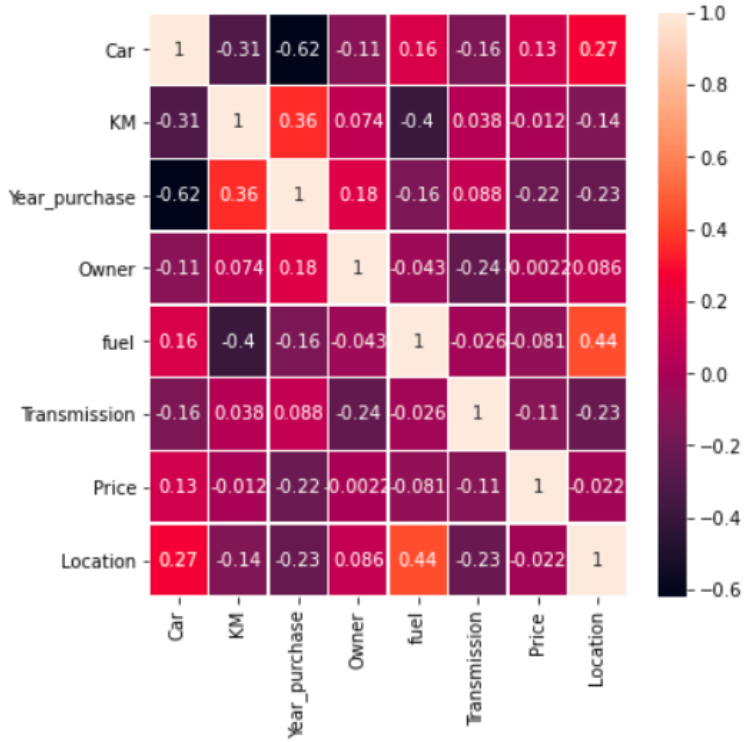
- Mean Squared Error
- Root Mean Squared Error
- Mean Absolute Error

Visualization

Checking the multi-collinearity. No major correlation found.

```
1 fig, ax = plt.subplots(figsize=(6,6))
2 sns.heatmap(df_encoded.corr(),annot=True, linewidth=0.5,)
```

```
<AxesSubplot:>
```



- **Interpretation of the Results**

  Selecting Linear Regression Model as it has low RMSE and better R2.

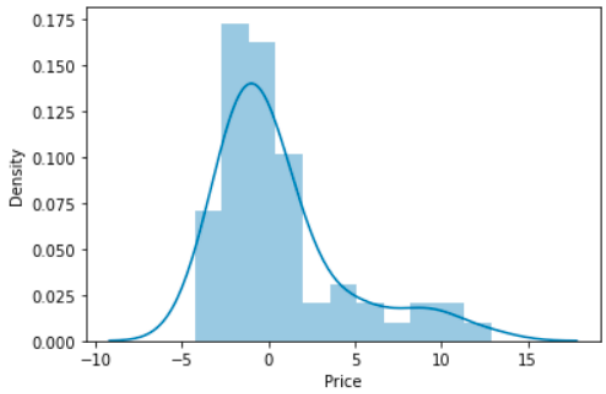
```
metrics of LinearRegression():
 mean_absolute_error: 2.6119050862281252
 mean_squared_error: 14.394900456087134
 r2_score: -0.03032565299391332
 mean cv score: -0.3301251089962346
```

CONCLUSION

- Key Findings and Conclusions of the Study

  Distribution plot: test data.

```
<AxesSubplot:xlabel='Price', ylabel='Density'>
```




```
1 results.sample(10)
```

:

	Actual	Predicted
201	5.47	6.678572
4	6.22	6.663167
123	2.74	5.699527
203	4.22	6.693762
162	4.68	5.492625
87	19.26	6.358141
175	4.76	5.261933
177	8.09	6.177097
169	2.87	5.601964
38	3.24	5.121539

- **Limitations of this work and Scope for Future Work**

  The scrapping of data is tedious task, and time taking process.

  Further if data scrapped is big in size, the result can be better.