FLIP ROBO

USED CAR PRICE PREDICTION PROJECT

Submitted by: Rajesh Kumar Singh

ACKNOWLEDGMENT

I would like to express my deepest gratitude to my SME (Subject Matter Expert) as well as Flip Robo Technologies who gave me the opportunity to do this project on Used Car Price Prediction, which also helped me in doing lots of research wherein I came to know about so many new things especially the data collection part.

Also, I have utilized a few external resources that helped me to complete the project. I ensured that I learn from the samples and modify things according to my project requirement. All the external resources that were used in creating this project are listed below:

1) https://www.google.com/
2) https://www.youtube.com/
3) https://github.com/
4) https://medium.com/
5) https://towardsdatascience.com/
6) https://www.analyticsvidhya.com/

INTRODUCTION

- Business Problem Framing

Impact of COVID-19 on Indian automotive sector: The Indian automotive sector was already struggling in FY20. before the Covid-19 crisis. It saw an overall degrowth of nearly 18 per cent. This situation was worsened by the onset of the Covid-19 pandemic and the ongoing lockdowns across India and the rest of the world. These two years (FY20 and FY21) are challenging times for the Indian automotive sector on account of slow economic growth, negative consumer sentiment, BS-VI transition, changes to the axle load norms, liquidity crunch, low-capacity utilisation and potential bankruptcies. The return of daily life and manufacturing activity to near normalcy in China and South Korea, along with extended lockdown in India, gives hope for a U-shaped economic recovery. Our analysis indicates that the Indian automotive sector will start to see recovery in the third quarter of FY21. We expect the industry demand to be down 15-25 per cent in FY21. With such degrowth, OEMs, dealers and suppliers with strong cash reserves and better access to capital will be better positioned to sail through. Auto sector has been under pressure due to a mix of demand and supply factors. However, there are also some positive outcomes, which we shall look at.

• With India's GDP growth rate for FY21 being downgraded from 5% to 0% and later to (-5%), the auto sector will take a hit. Auto demand is highly sensitive to job creation and income levels and both have been impacted. CII has estimated the revenue impact at $2 billion on a monthly basis across the auto industry in India.

• Supply chain could be the worst affected. Even as China recovers, supply chain disruptions are likely to last for some more time. The problems on the Indo-China border at Ladakh are not helping matters. Domestic suppliers are chipping in but they will face an inventory surplus as demand remains tepid.

• The Unlock 1.0 will coincide with the implementation of the BS-VI norms and that would mean heavier discounts to dealers and also to customers. Even as auto companies are managing costs, the impact of discounts on profitability is going to be fairly steep.

• The real pain could be on the dealer end with most of them struggling with excess inventory and lack of funding options in the post COVID-19 scenario. The BS-VI price increases are also likely to hit auto demand. There are two positive developments emanating from COVID-19. The China supply chain shock is forcing major investments in the "Make in India" initiative. The COVID-19 crisis has exposed chinks in the automobile business model and it could catalyse a big move towards electric vehicles (EVs). That could be the big positive for auto sector.

- ## Conceptual Background of the Domain Problem

The growing world of e-commerce is not just restricted to buying electronics and clothing but everything that you expect in a general store. Keeping the general store perspective aside and looking at the bigger picture, every day there are thousands or perhaps millions of deals happening in the digital marketplace. One of the most booming markets in the digital space is that of the automobile industry wherein the buying and selling of used cars take place. Sometimes we need to walk up to the dealer or individual sellers to get a used car price quote. However, buyers and sellers face a major stumbling block when it comes to their used car valuation or say their second-hand car valuation. Traditionally, you would go to a showroom and get your vehicle inspected before learning about the price. So instead of doing all these stuffs we can build a machine learning model using different features of the used cars to predict the exact and valuable car price.

- ## Review of Literature

This project is more about exploration, feature engineering and classification that can be done on this data. Since we scrape huge amount of data that includes more car related features, we can do better data exploration and derive some interesting features using the available columns.

The goal of this project is to build an application which can predict the car prices with the help of other features. In the long term, this would allow people to better explain and reviewing their purchase with each other in this increasingly digital world.

- ## Motivation for the Problem Undertaken

Based on the problem statement and the real time data scrapped from the OLX and Cars24 websites, I have understood how each independent feature helped me to understand the data as each feature provides a different kind of information. It is so interesting to work with different types of real time data in a single data set and perform root cause analysis to predict the price of the used car. Based on the analysis of the model of the car, kilometres driven, transmission type, fuel type etc. I would be able to model the price of used car as this model will then be used by the client to understand how exactly the prices vary with the variables. They can accordingly work on it and make some strategies to sell the used car and get some high returns. Furthermore, the model will be a good way for the client to understand the pricing dynamics of a used car.

Analytical Problem Framing

- Mathematical/ Analytical Modeling of the Problem

In our scrapped dataset, our target variable "Used Car Price " is a continuous variable. Therefore, we will be handling this modelling problem as regression.

This project is done in two parts:

- Data Collection phase
- Model Building phase

Data Collection phase:

You have to scrape at least 5000 used cars data. You can scrape more data as well, it's up to you. More the data better the model. In this section You need to scrape the data of used cars from websites (OLX, OLA, Car Dekho, Cars24 etc.) You need web scraping for this. You have to fetch data for different locations. The number of columns for data doesn't have limit, it's up to you and your creativity. Generally, these columns are Brand, model, variant, manufacturing year, driven kilometres, fuel, number of owners, location and at last target variable Price of the car. This data is to give you a hint about important variables in used car model. You can make changes to it, you can add or you can remove some columns, it completely depends on the website from which you are fetching the data. Try to include all types of cars in your data for example- SUV, Sedans, Coupe, minivan, Hatchback.

Model Building phase:

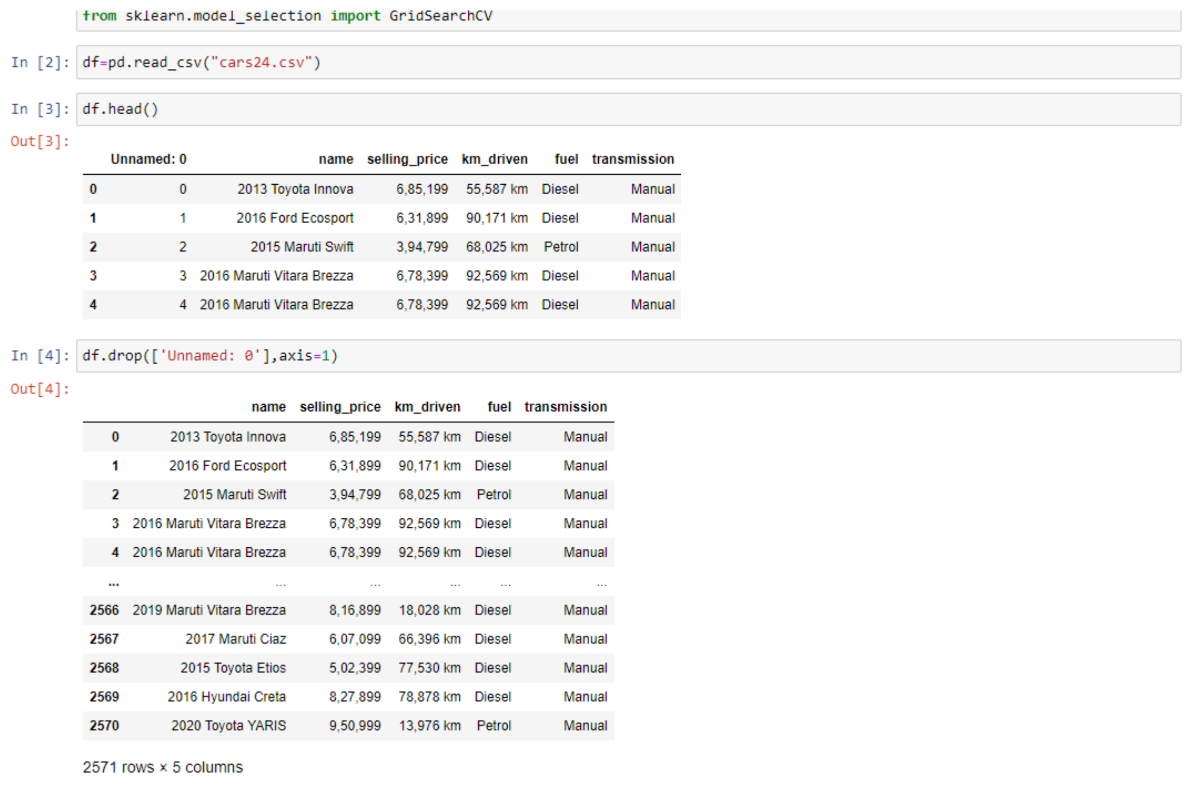
After collecting the data, you need to build a machine learning model. Before model building do all data pre-processing steps. Try different models with different hyper parameters and select the best model. Follow the complete life cycle of data science. Include all the below steps mentioned:

1. Data Cleaning
2. Exploratory Data Analysis (EDA)
3. Data Pre-processing and Visualisation
4. Model Building
5. Model Evaluation
6. Selecting the best model

- ## Data Sources and their formats

The dataset is in the form of CSV (Comma Separated Value) format and consists of 6 columns (5 features and 1 label) with 2700 number of records as explained below:

- Names - This shows the car model names
- Selling_price - Gives us the selling price of the car.
- Km_Driven - Number of kilometres the car the driven reflecting on the Odometer
- Fuel - Shows the fuel type used by the vehicle
- Transmission - Gives us the manual or automatic gear shifting mechanism



```
from sklearn.model_selection import GridSearchCV
```

```
In [2]: df=pd.read_csv("cars24.csv")
```

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

Out[3]:

	Unnamed: 0	name	selling_price	km_driven	fuel	transmission
0	0	2013 Toyota Innova	6,85,199	55,587 km	Diesel	Manual
1	1	2016 Ford Ecosport	6,31,899	90,171 km	Diesel	Manual
2	2	2015 Maruti Swift	3,94,799	68,025 km	Petrol	Manual
3	3	2016 Maruti Vitara Brezza	6,78,399	92,569 km	Diesel	Manual
4	4	2016 Maruti Vitara Brezza	6,78,399	92,569 km	Diesel	Manual

```
In [4]: df.drop(['Unnamed: 0'],axis=1)
```

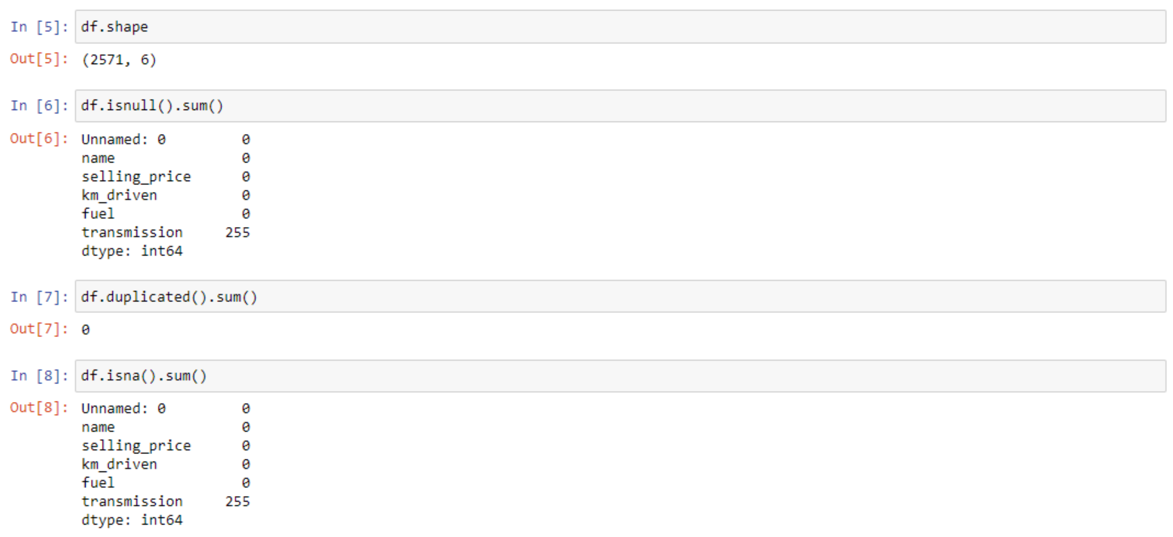
Out[4]:

	name	selling_price	km_driven	fuel	transmission
0	2013 Toyota Innova	6,85,199	55,587 km	Diesel	Manual
1	2016 Ford Ecosport	6,31,899	90,171 km	Diesel	Manual
2	2015 Maruti Swift	3,94,799	68,025 km	Petrol	Manual
3	2016 Maruti Vitara Brezza	6,78,399	92,569 km	Diesel	Manual
4	2016 Maruti Vitara Brezza	6,78,399	92,569 km	Diesel	Manual
...	...	...	...	...	...
2566	2019 Maruti Vitara Brezza	8,16,899	18,028 km	Diesel	Manual
2567	2017 Maruti Ciaz	6,07,099	66,396 km	Diesel	Manual
2568	2015 Toyota Etios	5,02,399	77,530 km	Diesel	Manual
2569	2016 Hyundai Creta	8,27,899	78,878 km	Diesel	Manual
2570	2020 Toyota YARIS	9,50,999	13,976 km	Petrol	Manual

2571 rows × 5 columns

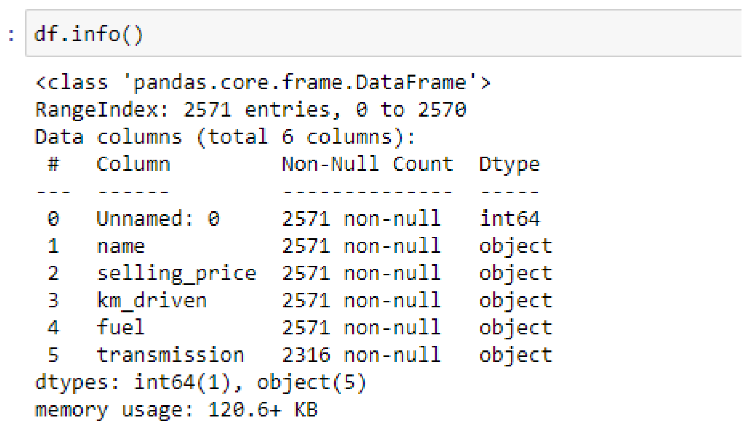
- ## Data Preprocessing Done

For the data pre-processing step, I checked through the dataframe for missing values, imputed records with "-" using various imputing techniques to handle them.



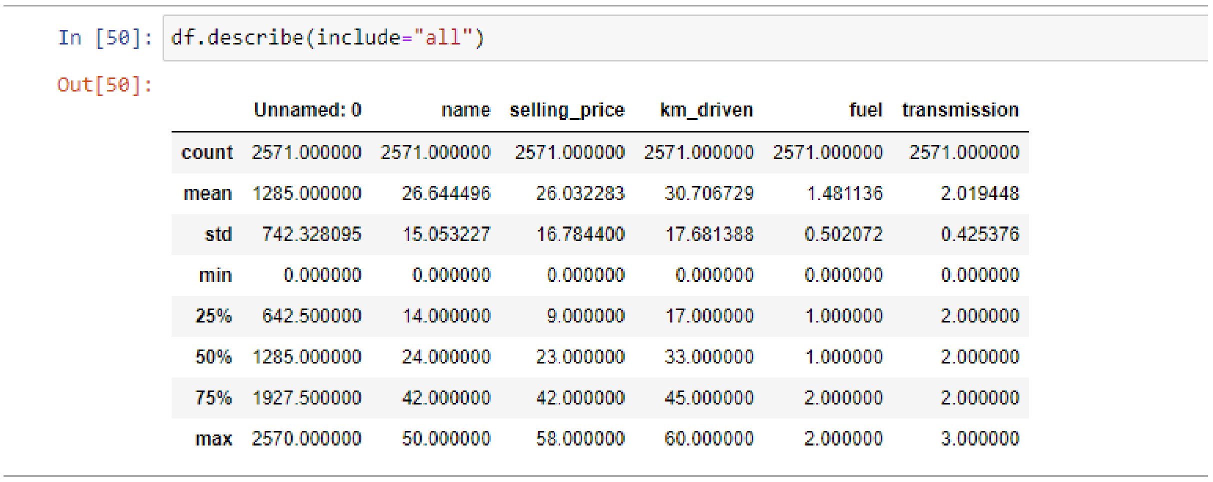
```
In [5]: df.shape

Out[5]: (2571, 6)

In [6]: df.isnull().sum()

Out[6]: Unnamed: 0         0
        name               0
        selling_price      0
        km_driven          0
        fuel               0
        transmission     255
        dtype: int64

In [7]: df.duplicated().sum()

Out[7]: 0

In [8]: df.isna().sum()

Out[8]: Unnamed: 0         0
        name               0
        selling_price      0
        km_driven          0
        fuel               0
        transmission     255
        dtype: int64
```

Checked the datatype details for each column to understand the numeric ones and its further conversion process.



```
: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2571 entries, 0 to 2570
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Unnamed: 0     2571 non-null   int64
 1   name           2571 non-null   object
 2   selling_price  2571 non-null   object
 3   km_driven      2571 non-null   object
 4   fuel           2571 non-null   object
 5   transmission   2316 non-null   object
dtypes: int64(1), object(5)
memory usage: 120.6+ KB
```

I then used the “describe” method to check the count, mean, standard deviation, minimum, maximum, 25%, 50% and 75% quartile data.

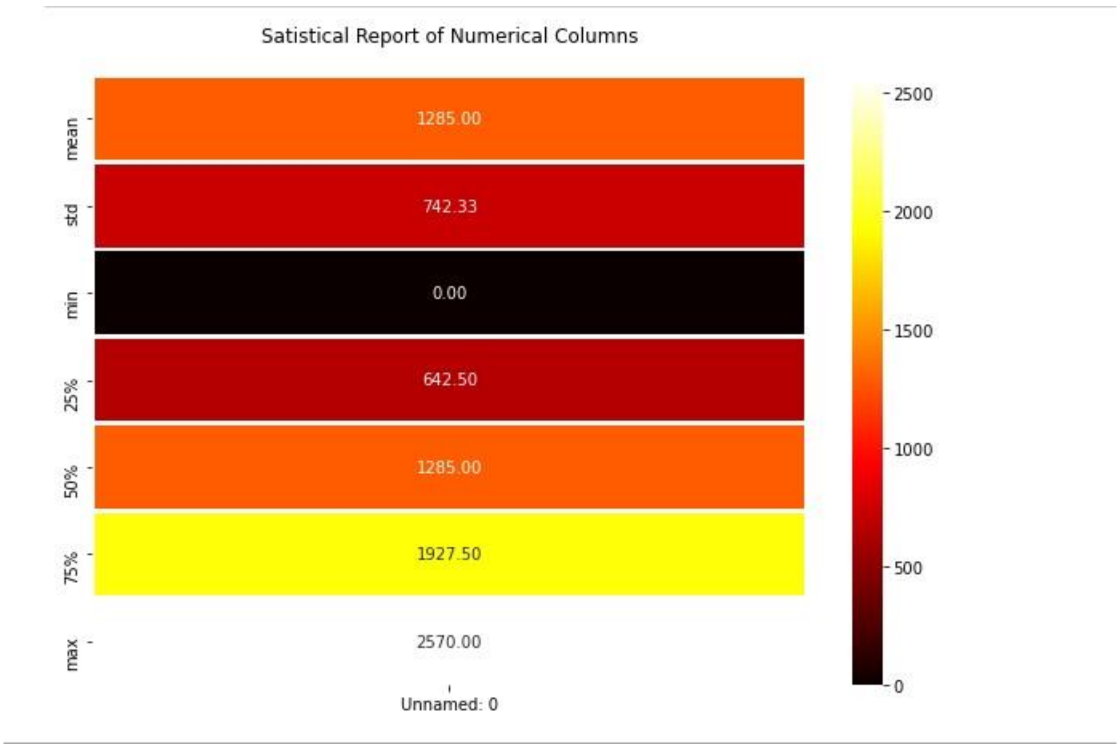


```
In [50]: df.describe(include="all")
```

Out[50]:

	Unnamed: 0	name	selling_price	km_driven	fuel	transmission
count	2571.000000	2571.000000	2571.000000	2571.000000	2571.000000	2571.000000
mean	1285.000000	26.644496	26.032283	30.706729	1.481136	2.019448
std	742.328095	15.053227	16.784400	17.681388	0.502072	0.425376
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	642.500000	14.000000	9.000000	17.000000	1.000000	2.000000
50%	1285.000000	24.000000	23.000000	33.000000	1.000000	2.000000
75%	1927.500000	42.000000	42.000000	45.000000	2.000000	2.000000
max	2570.000000	50.000000	58.000000	60.000000	2.000000	3.000000

Took a visual on just the numeric part as well and saw just the maximum value for Used Car Price column at a higher scale.



- **Hardware and Software Requirements and Tools Used**

➢ <u>Hardware technology being used.</u>

RAM	: 4 GB
CPU	: INTEL Core i3, 1.99GHz.
GPU	: NVIDIA GETFORCE.

➢ <u>Software technology being used.</u>

Programming language	: Python
Distribution	: Anaconda Navigator
Browser based language shell	: Jupyter Notebook

➢ <u>Libraries/Packages specifically being used.</u>

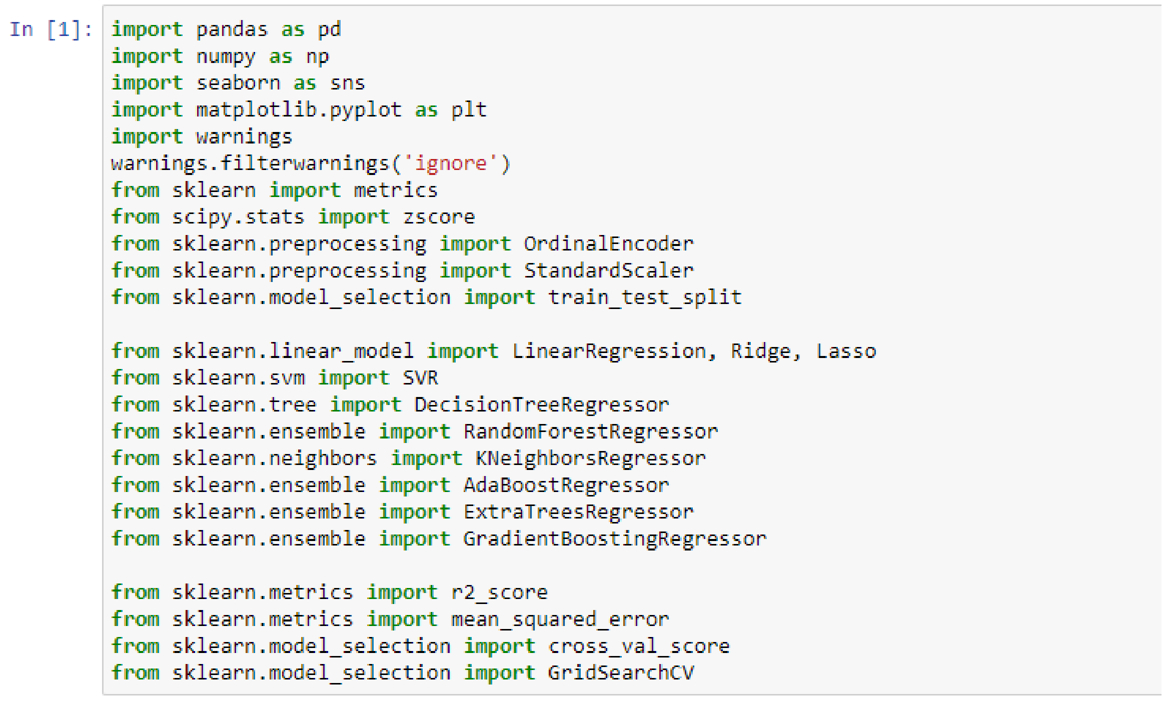
<u>Pandas, NumPy, matplotlib, seaborn.</u>

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)
  1. Clean the dataset from unwanted scraped details.
  2. Impute missing values with meaningful information.
  3. Encoding the categorical data to get numerical input data.
  4. Compare different models and identify the suitable model.
  5. R2 score is used as the primary evaluation metric.
  6. MSE and RMSE are used as secondary metrics.
  7. Cross Validation Score was used to ensure there are no overfitting our underfitting models.

- Testing of Identified Approaches (Algorithms)

  Libraries and Machine Learning Regression models that were used in this project are shown below.



```
In [1]: import pandas as pd
        import numpy as np
        import seaborn as sns
        import matplotlib.pyplot as plt
        import warnings
        warnings.filterwarnings('ignore')
        from sklearn import metrics
        from scipy.stats import zscore
        from sklearn.preprocessing import OrdinalEncoder
        from sklearn.preprocessing import StandardScaler
        from sklearn.model_selection import train_test_split

        from sklearn.linear_model import LinearRegression, Ridge, Lasso
        from sklearn.svm import SVR
        from sklearn.tree import DecisionTreeRegressor
        from sklearn.ensemble import RandomForestRegressor
        from sklearn.neighbors import KNeighborsRegressor
        from sklearn.ensemble import AdaBoostRegressor
        from sklearn.ensemble import ExtraTreesRegressor
        from sklearn.ensemble import GradientBoostingRegressor

        from sklearn.metrics import r2_score
        from sklearn.metrics import mean_squared_error
        from sklearn.model_selection import cross_val_score
        from sklearn.model_selection import GridSearchCV
```

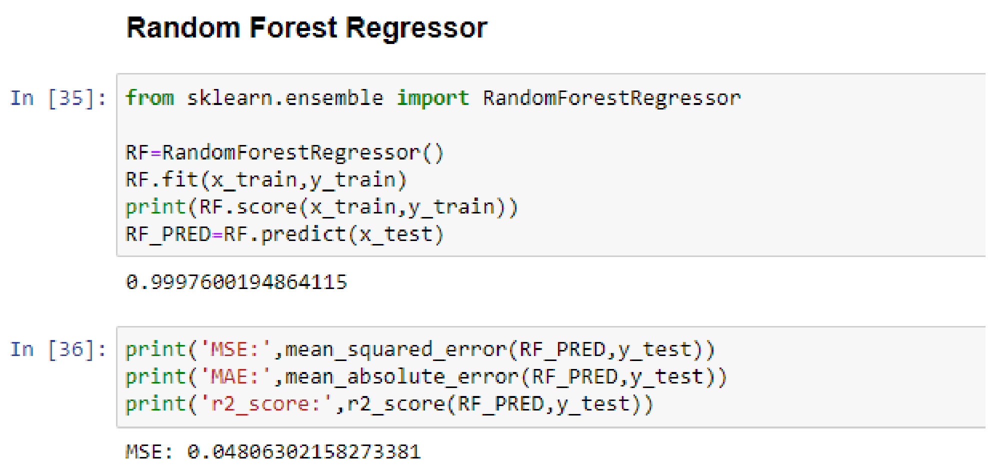
All the regression machine learning algorithms used are:

- Linear Regression Model
- Ridge Regularization Model
- Lasso Regularization Model
- Support Vector Regression Model
- Decision Tree Regression Model
- Random Forest Regression Model
- K Neighbours Regression Model
- Gradient Boosting Regression Model
- Ada Boost Regression Model
- Extra Trees Regression Model

- ## Run and Evaluate selected models

I created a Regression Machine Leaning Model function incorporating the evaluation metrics so that we can get the required data for all the above models.

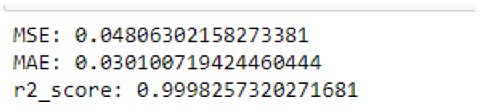
Code:



Random Forest Regressor

```
In [35]: from sklearn.ensemble import RandomForestRegressor

         RF=RandomForestRegressor()
         RF.fit(x_train,y_train)
         print(RF.score(x_train,y_train))
         RF_PRED=RF.predict(x_test)

         0.9997600194864115

In [36]: print('MSE:',mean_squared_error(RF_PRED,y_test))
         print('MAE:',mean_absolute_error(RF_PRED,y_test))
         print('r2_score:',r2_score(RF_PRED,y_test))

         MSE: 0.04806302158273381
```

Output:



```
MSE: 0.04806302158273381
MAE: 0.030100719424460444
r2_score: 0.9998257320271681
```

- Key Metrics for success in solving problem under consideration

RMSE Score:
Root Mean Square Error (RMSE) is the standard deviation of the residuals (prediction errors). Residuals are a measure of how far from the regression line data points are; RMSE is a measure of how spread out these residuals are. In other words, it tells you how concentrated the data is around the line of best fit.

R2 Score:
The R2 score is a very important metric that is used to evaluate the performance of a regression-based machine learning model. It is pronounced as R squared and is also known as the coefficient of determination. It works by measuring the amount of variance in the predictions explained by the dataset.

Cross Validation Score:
Cross-validation is a statistical method used to estimate the skill of machine learning models. It is commonly used in applied machine learning to compare and select a model for a given predictive modelling problem because it is easy to understand, easy to implement, and results in skill estimates that generally have a lower bias than other methods. The k-fold cross validation is a procedure used to estimate the skill of the model on new data. There are common tactics that you can use to select the value of k for your dataset (I have used 5-fold validation in this project). There are commonly used variations on cross-validation such as stratified and repeated that are available in scikit-learn.

Hyper Parameter Tuning:
In machine learning, hyperparameter optimization or tuning is the problem of choosing a set of optimal hyperparameters for a learning algorithm. A hyperparameter is a parameter whose value is

used to control the learning process. By contrast, the values of other parameters (typically node weights) are learned.
Code:

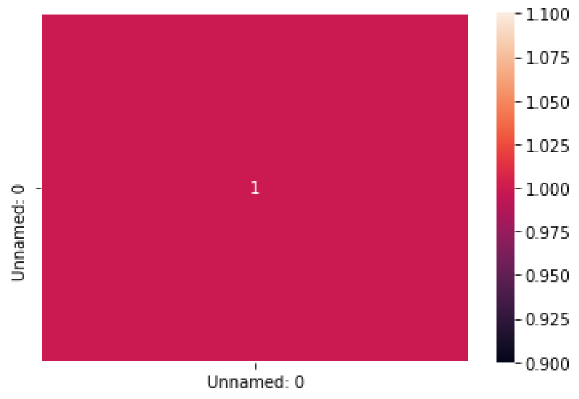
Final model score after plugging in the best parameter values:

- Visualizations

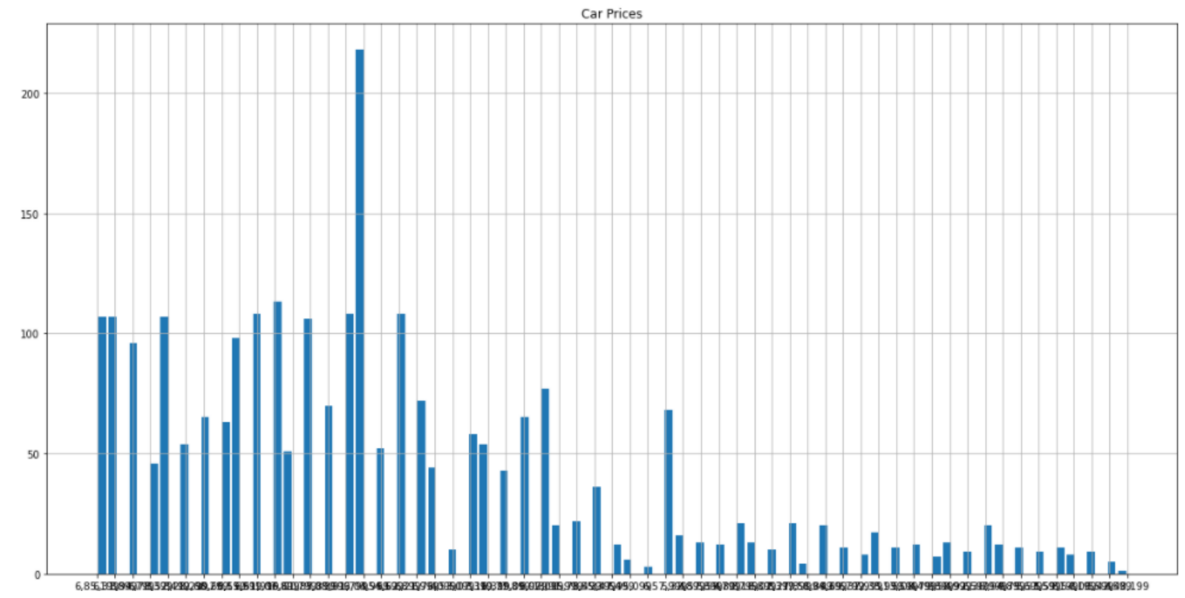
I generated count plots, bar plots, pair plots, heatmap and others to visualise the datapoint present in our column records.

Code:

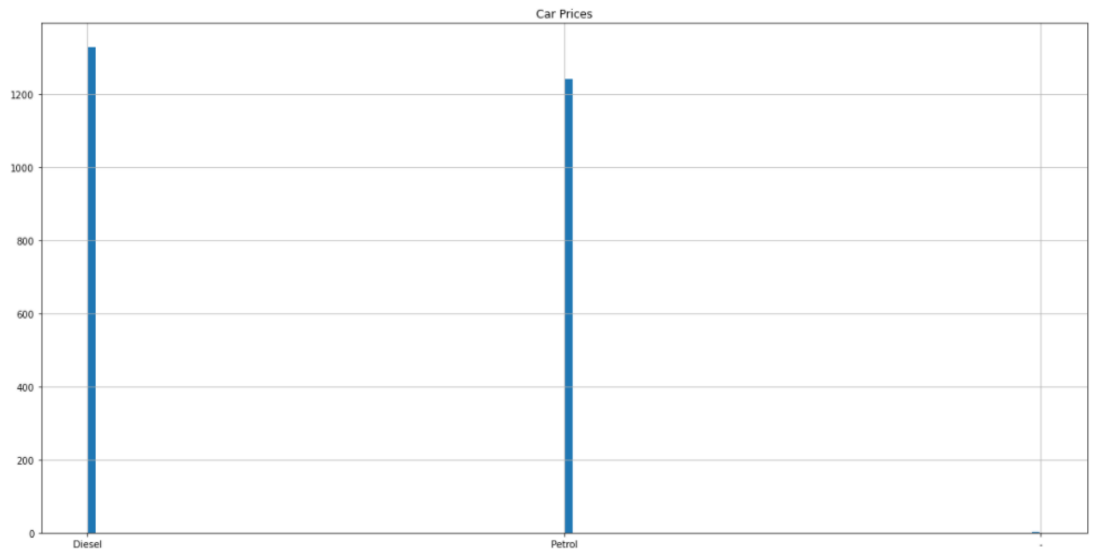
```
In [13]: sns.heatmap(dfcorr,annot=True, linewidth=1)

Out[13]: <AxesSubplot:>
```



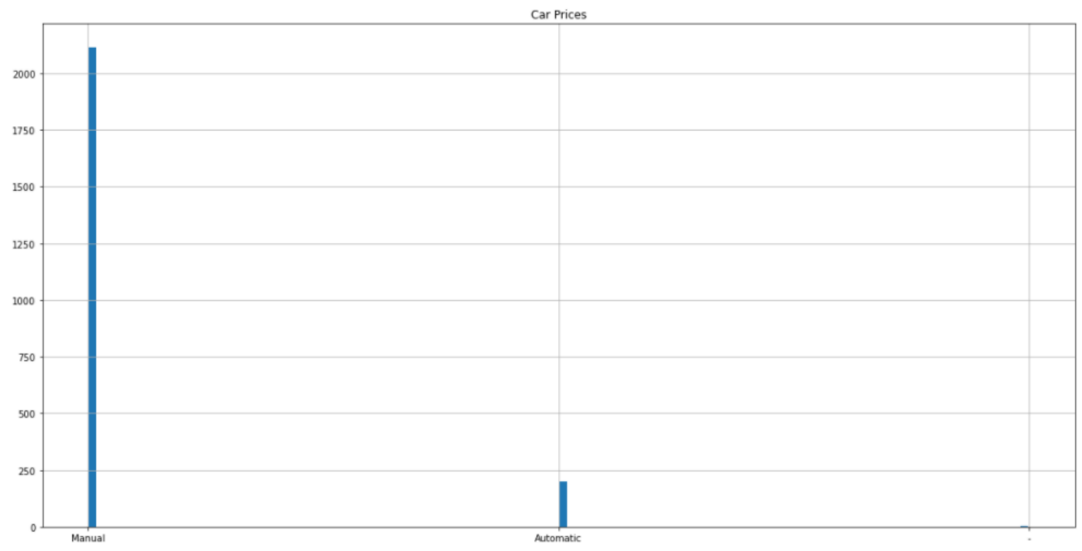
```
: df['selling_price'].hist(bins=100,figsize=(20,10),rwidth = 0.75)
  plt.title('Car Prices')
  plt.show()
```



```
In [15]: df['fuel'].hist(bins=100,figsize=(20,10),rwidth = 0.75)
         plt.title('Car Prices')
         plt.show()
```

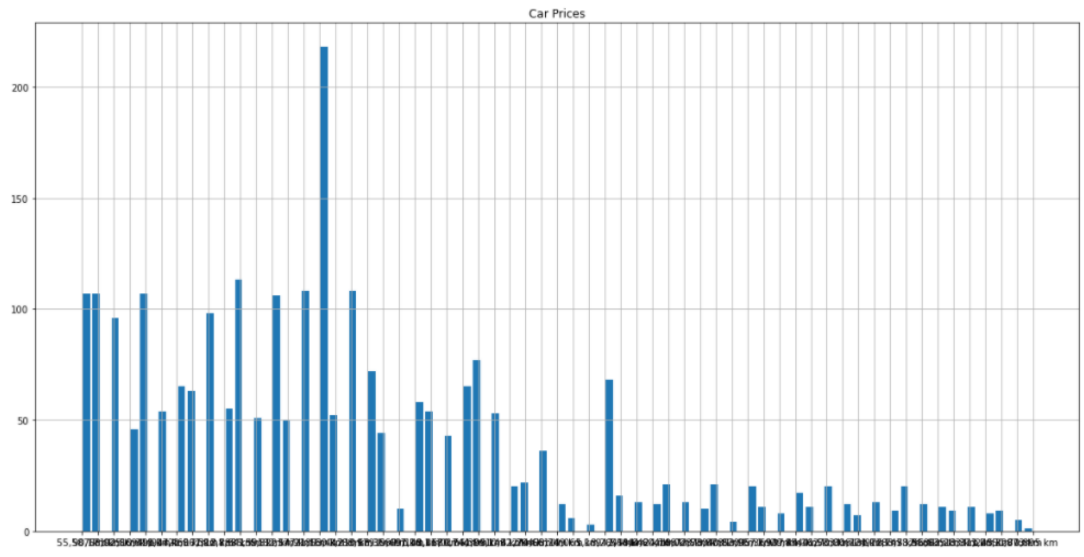


```
In [16]: df['transmission'].hist(bins=100,figsize=(20,10),rwidth = 0.75)
         plt.title('Car Prices')
         plt.show()
```



In [17]: df['km_driven'].hist(bins=100,figsize=(20,10),rwidth = 0.75)

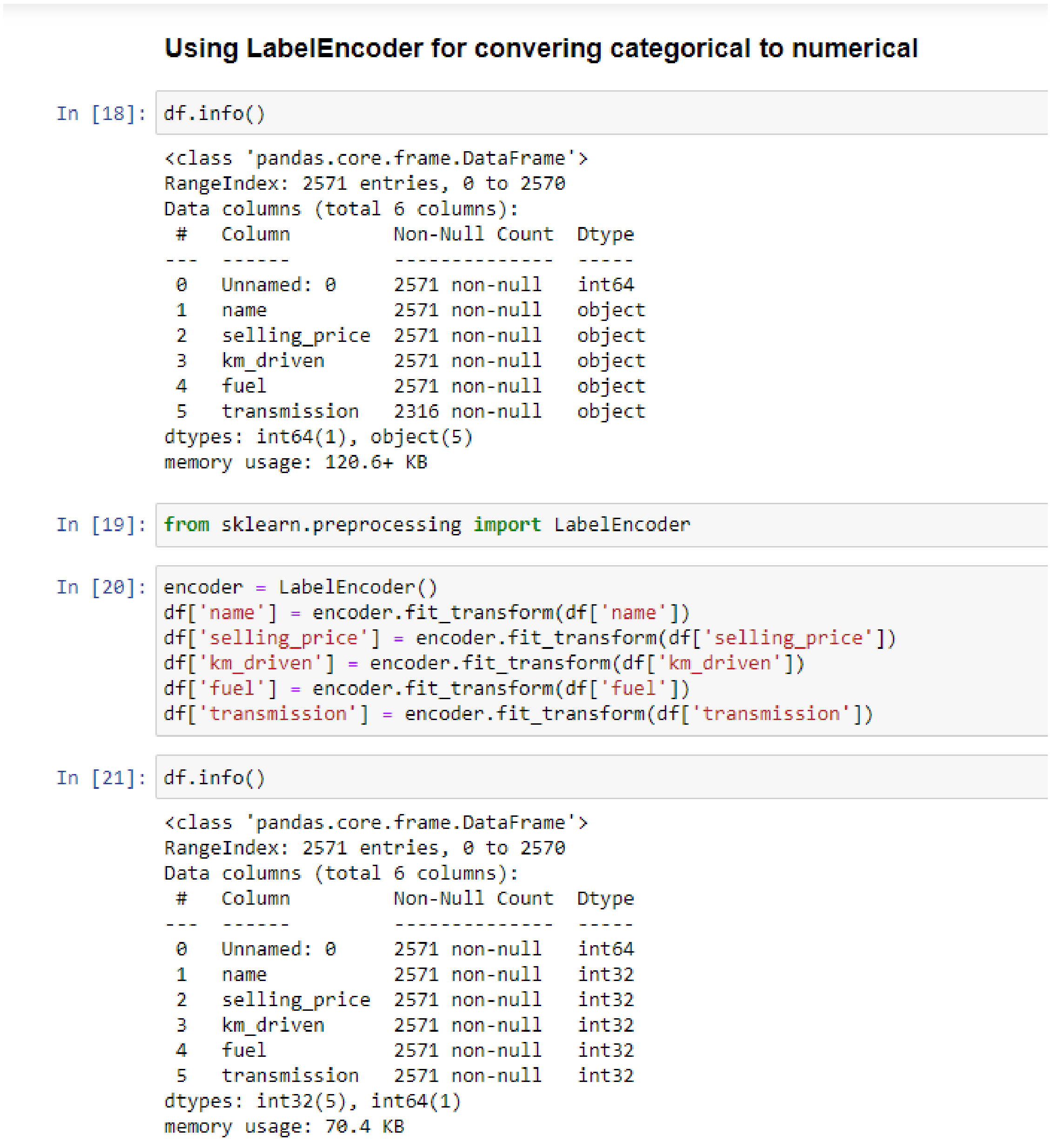
```
In [17]: df['km_driven'].hist(bins=100,figsize=(20,10),rwidth = 0.75)
         plt.title('Car Prices')
         plt.show()
```



:

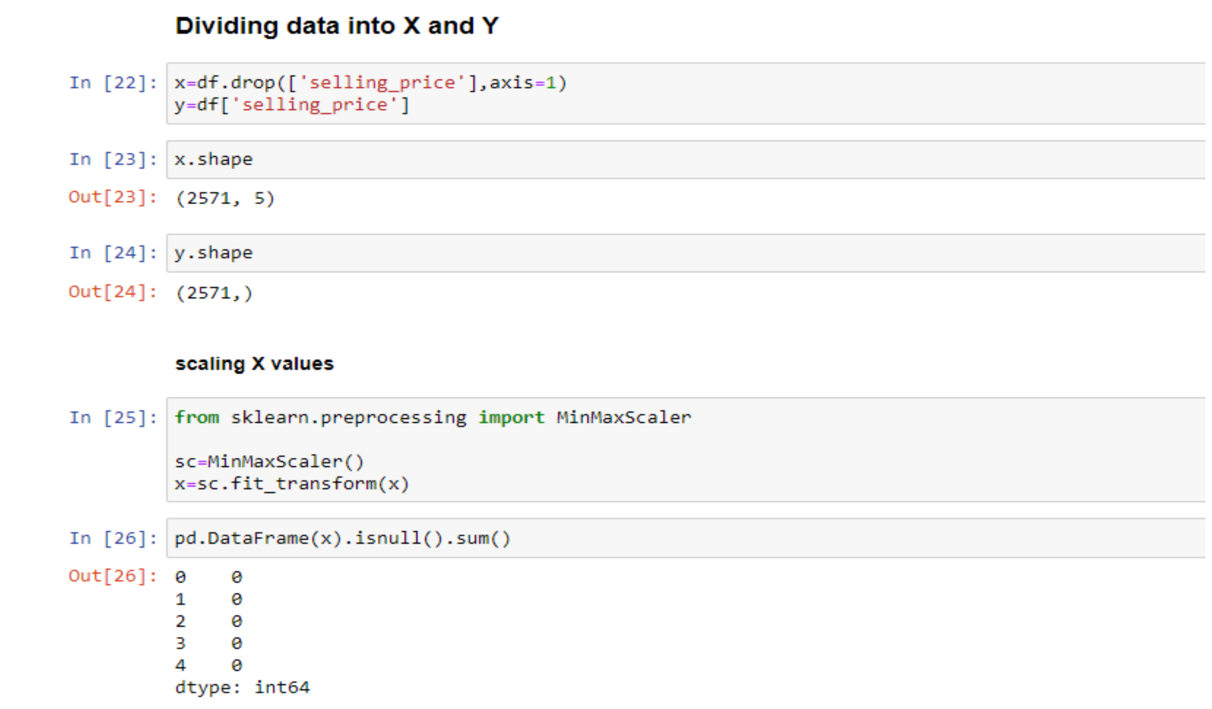
- Interpretation of the Results

We can see from the visuals that the features are impacting the price of used cars. There were categorical columns which I encoded using the ordinal encoder method instead of the one hot encoding to avoid the generation of large number of columns. Also, I our target label stored continuous numeric data and therefore label encoder was out of the picture to be used.



Using LabelEncoder for convering categorical to numerical

```
In [18]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2571 entries, 0 to 2570
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   Unnamed: 0     2571 non-null   int64 
 1   name           2571 non-null   object
 2   selling_price  2571 non-null   object
 3   km_driven      2571 non-null   object
 4   fuel           2571 non-null   object
 5   transmission   2316 non-null   object
dtypes: int64(1), object(5)
memory usage: 120.6+ KB

In [19]: from sklearn.preprocessing import LabelEncoder

In [20]: encoder = LabelEncoder()
         df['name'] = encoder.fit_transform(df['name'])
         df['selling_price'] = encoder.fit_transform(df['selling_price'])
         df['km_driven'] = encoder.fit_transform(df['km_driven'])
         df['fuel'] = encoder.fit_transform(df['fuel'])
         df['transmission'] = encoder.fit_transform(df['transmission'])

In [21]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2571 entries, 0 to 2570
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Unnamed: 0     2571 non-null   int64
 1   name           2571 non-null   int32
 2   selling_price  2571 non-null   int32
 3   km_driven      2571 non-null   int32
 4   fuel           2571 non-null   int32
 5   transmission   2571 non-null   int32
dtypes: int32(5), int64(1)
memory usage: 70.4 KB
```



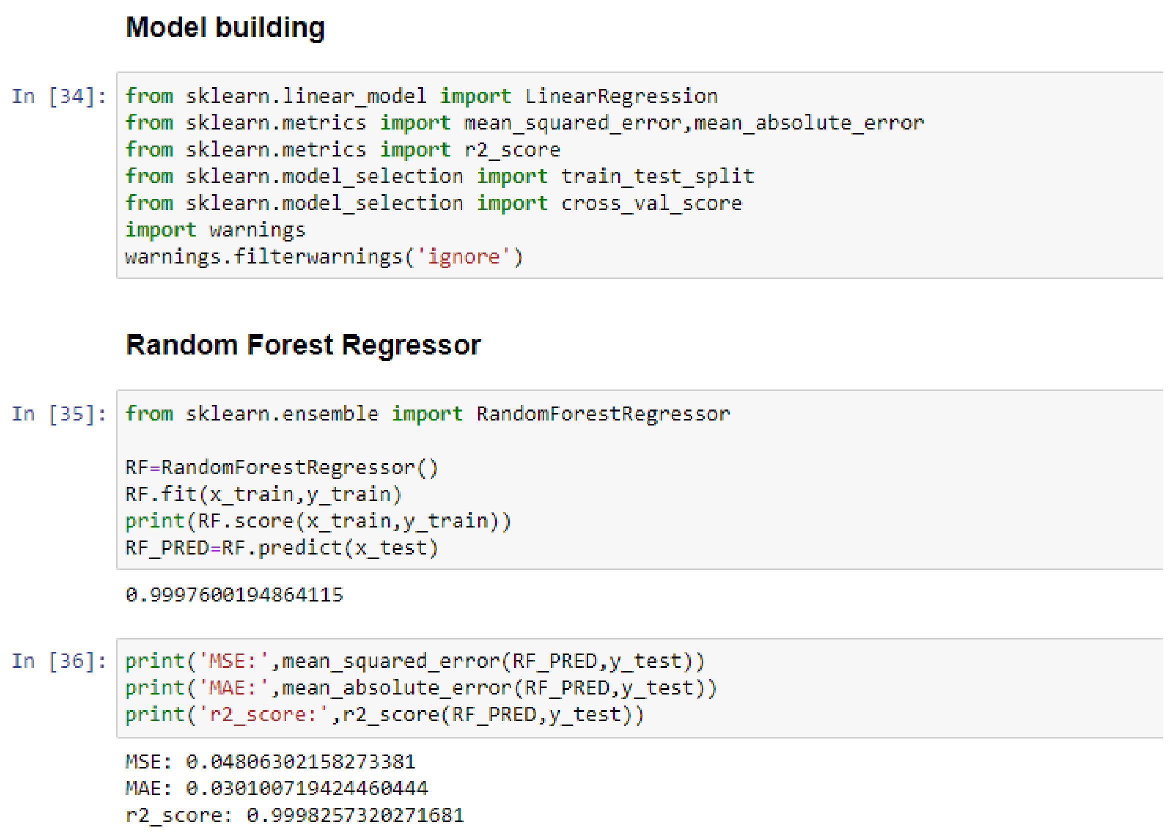
Dividing data into X and Y

```
In [22]: x=df.drop(['selling_price'],axis=1)
         y=df['selling_price']

In [23]: x.shape

Out[23]: (2571, 5)

In [24]: y.shape

Out[24]: (2571,)
```

scaling X values

```
In [25]: from sklearn.preprocessing import MinMaxScaler

         sc=MinMaxScaler()
         x=sc.fit_transform(x)

In [26]: pd.DataFrame(x).isnull().sum()

Out[26]: 0    0
         1    0
         2    0
         3    0
         4    0
         dtype: int64
```

<u>Model building and saving the model</u>

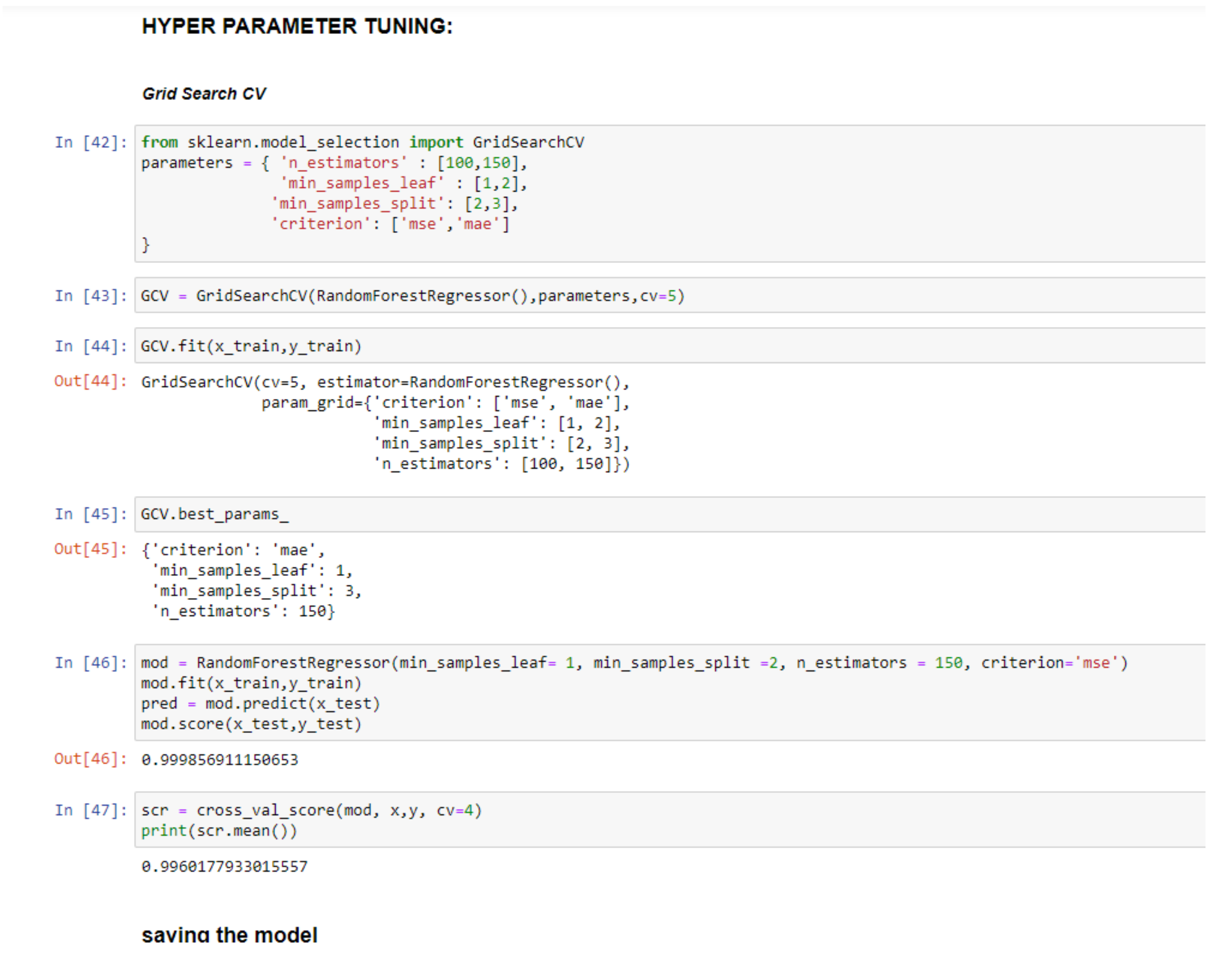


Model building

```
In [34]: from sklearn.linear_model import LinearRegression
         from sklearn.metrics import mean_squared_error,mean_absolute_error
         from sklearn.metrics import r2_score
         from sklearn.model_selection import train_test_split
         from sklearn.model_selection import cross_val_score
         import warnings
         warnings.filterwarnings('ignore')
```

Random Forest Regressor

```
In [35]: from sklearn.ensemble import RandomForestRegressor

         RF=RandomForestRegressor()
         RF.fit(x_train,y_train)
         print(RF.score(x_train,y_train))
         RF_PRED=RF.predict(x_test)

         0.9997600194864115

In [36]: print('MSE:',mean_squared_error(RF_PRED,y_test))
         print('MAE:',mean_absolute_error(RF_PRED,y_test))
         print('r2_score:',r2_score(RF_PRED,y_test))

         MSE: 0.04806302158273381
         MAE: 0.030100719424460444
         r2_score: 0.9998257320271681
```



HYPER PARAMETER TUNING:

Grid Search CV

```
In [42]: from sklearn.model_selection import GridSearchCV
         parameters = { 'n_estimators' : [100,150],
                        'min_samples_leaf' : [1,2],
                       'min_samples_split': [2,3],
                       'criterion': ['mse','mae']
         }

In [43]: GCV = GridSearchCV(RandomForestRegressor(),parameters,cv=5)

In [44]: GCV.fit(x_train,y_train)

Out[44]: GridSearchCV(cv=5, estimator=RandomForestRegressor(),
                      param_grid={'criterion': ['mse', 'mae'],
                                  'min_samples_leaf': [1, 2],
                                  'min_samples_split': [2, 3],
                                  'n_estimators': [100, 150]})

In [45]: GCV.best_params_

Out[45]: {'criterion': 'mae',
          'min_samples_leaf': 1,
          'min_samples_split': 3,
          'n_estimators': 150}

In [46]: mod = RandomForestRegressor(min_samples_leaf= 1, min_samples_split =2, n_estimators = 150, criterion='mse')
         mod.fit(x_train,y_train)
         pred = mod.predict(x_test)
         mod.score(x_test,y_test)

Out[46]: 0.999856911150653

In [47]: scr = cross_val_score(mod, x,y, cv=4)
         print(scr.mean())

         0.9960177933015557
```

saving the model

CONCLUSION

- Key Findings and Conclusions of the Study

  After the completion of this project, we got an insight on how to collect data, pre-processing the data, analysing the data and building a model. First, we collected the used cars data from different websites like OLX, Car Dekho, Cars 24, OLA etc and it was done by using Web Scraping. The framework used for web scraping was Beautiful Soup and Selenium, which has an advantage of automating our process of collecting data. We collected almost 10000 of data which contained the selling price and other related features of used cars. Then the scrapped data was combined in a single data frame and saved in a csv file so that we can open it and analyse the data. We did data cleaning, data pre-processing steps like finding and handling null values, removing words from numbers, converting object to int type, data visualization, handling outliers and skewness etc. After separating our train and test data, we started running different machine learning regression algorithms to find out the best performing model. We found that Extra Tree Regressor Algorithm was performing well according to their r2_score and cross validation scores. Then we performed Hyperparameter Tuning technique using Grid Search CV for getting the best parameters and improving the score. In that Extra Tree Regressor Algorithm did not perform quite well as previously on the defaults but we finalised that model for further predictions as it was still better than the rest. We saved the final model in pkl format using the joblib library after getting a dataframe of predicted and actual used car price details.

- Learning Outcomes of the Study in respect of Data Science

  Visualization part helped me to understand the data as it provides graphical representation of huge data. It assisted me to understand the feature importance, outliers/skewness detection and to compare the independent-dependent features. Data cleaning is the most important part of model building and therefore before model building, I made sure the data is cleaned and scaled. I have generated multiple regression machine learning models to get the best model wherein I found Extra Trees Regressor Model being the best based on the metrics I have used.

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

The limitations we faced during this project were:

The website was poorly designed because the scrapping took a lot of time and there were many issues in accessing to next page. Also need further practise in terms of various web scraping techniques. More negative correlated data were present than the positive correlated one's. Presence of outliers and skewness were detected and while dealing with them we had to lose a bit of valuable data. No information for handling these fast-paced websites were provided so that was consuming more time in web scraping part.

Future Work Scope:

Current model is limited to used car data but this can further be improved for other sectors of automobiles by training the model accordingly. The overall score can also be improved further by training the model with more specific data.