# Major Project On

# Introduction to Big-Data

**Predicting the cost of used cars from given the data**

Sarthak Giri
Abhipsa Panda
Ashwin Raj
Muskan Khurana
Sneha Sreelata
Chitrabanu
Anshul Srivastava
Simarpreet Kaur

## 1.0 Objective

Predicting the costs of used cars given the data collected from various sources and distributed across various locations in India.Data contains the variables: Name, Location, Year, Kilometers Driven, FuelType, Transmission, Owner Type, Mileage, Engine, Power, Seats, Price

### Description of Independent and Dependent Variables:

### Independent Variables:

| | |
|---|---|
| **Name:** | The brand and model of the car. |
| **Location:** | The location in which the car is being sold or is available for purchase. |
| **Year:** | The year or edition of the model. |
| **Kilometers_Driven:** | The total kilometers driven in the car by the previous owner(s) in KM. |
| **Fuel_Type:** | The type of fuel used by the car. |
| **Transmission:** | The type of transmission used by the car. |
| **Owner_Type:** | Whether the ownership is Firsthand, Second hand, or other. |
| **Mileage:** | The standard mileage offered by the car company in kmpl or km/kg |
| **Engine:** | The displacement volume of the engine in cc. |
| **Power:** | The maximum power of the engine in bhp. |
| **Seats:** | The number of seats in the car. |
| **New_Price:** | The price of a new car of the same model. |

### Dependent Variables:

| | |
|---|---|
| **Price:** | The price of the used car in lakhs INR. |

## 2.0 **Introduction**

Data was given in a '.csv' file. It was analyzed and an algorithmic ML model has been made using Python libraries of 'Pandas', 'Numpy' and 'Sklearn', whereas 'Seaborn' and 'Matplotlib.pyplot' were used for creating graphs for data exploration. The outputs are presented in terms of insight. The python program along with the output is also enclosed as an appendix to this report.

## 3.0 **Cleaning the data**

The Data Cleaning part of EDA has been done by Ashwin Raj. The null values in the data were suitably dealt with, with the final, clean data set having no null values.

## 4.0 **Data Exploration**

As mentioned above, 'Seaborn' and 'Matplotlib.pyplot' were used for creating exploratory graphs for the data, and 'Pandas' was used as well for importing the data set into python and to help with exploring the data and deriving insights necessary for deciding which variables to use, which models will be best to be used for predicting prices, how to convert the categorical variables into numerical variables, and how to scale the numerical variables.

The list below has the name of people and the variables they have worked on, for Exploratory Data Analysis:

1. Sarthak Giri- Mileage, Engine
2. Abhipsa Panda- Location, Owner_Type
3. Ashwin Raj- Kilometers_driven
4. Muskan Khurana- Year
5. Sneha Sreelata- Transmission, Seats
6. Chitrabanu- Power
7. Anshul- Fuel_Type (and all other variables)

## 5.0 **ML Model Information**

All information about the Machine Learning model to be used and the related steps are shown below.

The variables in the train data set can be considered as follows:

1. Name: We can divide this variable into 3 more variables (Brand, Car_Name, Model), and use label encoding or one-hot encoding to create categorical data points.
2. Location: We can create categorical data points in this variable using one-hot encoding or label encoding.
3. Year: We can change it into an ordinal variable, using label encoding or one-hot encoding.
4. Fuel_Type: One-hot encoding or label encoding can be used to convert it into a categorical variable.
5. Owner_Type: Use label encoding or one-hot encoding to convert it to a categorical variable.
6. Transmission: Use label encoding or one-hot encoding to convert it to a categorical variable.

The important variables are:

```
Name, Location, Year, Fuel-Type, Engine, Power, Seats, Owner-Type, Transmission
```

The variables that are not important:

```
Kilometers_Driven, Mileage
```

Error Metrics that have been used:
R^2, Adjusted R^2, RMSE

## 6.0 **Methods used to train the model**

The Methods used to clean the data and train the model:

1. Converted the Year variable into a label encoded variable (ordinal), and converted the Location variable into a one-hot encoded variable. As of now, the data set has 21 columns, since the 10 unique city-name variables were created from the Location variable, and the original Location variable was removed.
2. In the Name variable, separated out the brand names, and converted them into one-hot encoded variables, creating 28 more columns, and the original Name variable was removed.
3. Created one-hot encoded variables out of the Fuel_Type variable, adding 3 more columns, and the original Fuel_Type column has been removed.
4. Created one-hot encoded variables out of the Seats variable, adding 7 more columns, and the original Seats column has been removed.
5. The Transmission variable has been one-hot encoded, which has created 1 more variable, and the original Transmission column has been removed.
6. The Owner_Type variable has been one-hot encoded, which has created 3 more columns, and the original Owner_Type variable has been removed.

Below is a list of people who have worked on the variables mentioned in the above 6 steps:

a) Sarthak Giri - Year, Location, Name, Seats
b) Simar Preet Kaur - Owner_Type, Fuel_Type
c) Anshul Srivastava - Transmission

(In all one-hot encoded variables, the first variable has been dropped, to prevent the problem of multicollinearity.)

7. Removed the Kilometers_Driven and the Mileage(kmpl) variables, since they don't affect the predicted price of a car in a practical scenario. The kilometers driven is an arbitrary variable, and there is no general relationship between the value of a car and the kilometers that it has been driven for. As for the mileage, that also is an arbitrary variable, and the kmpl or km/kg mileage rating of a car is an approximate value and is not really sought after by a car buyer.

8. Feature Scaled the Engine(CC) and the Power(bhp) variable, either using min-max scaling or standard scaling.
    - Robust Scaling has been used to scale the values in these two variables.

9. Split the training data further into train and test data sets (probably a 70-30% split).

10. Trained the model on the training data (from the train-test split), validate it using k-folds cross-validation, then apply the trained and validated model on the test data (from the train-test split), and calculate the error metrics. (The error metrics that need to be calculated are mentioned here: ML Models Information).
    - The model used for training and testing is Linear Regression

## 7.0 **Conclusion**:

After splitting the training data set into training and test sets, and applying the linear regression model on the training set, we got these error metrics:

1. R Square Score: `0.779775569159032`
2. Adjusted R Square Score: `0.7726757193956948`

   Both the r-square and adjusted r-square values show that the model explains a good amount of the variance on the data.

3. Root Mean Square Error: `5.17564264835917 Lakh INR`

   The low value of RMSE indicates that this model has a good prediction power (lower the RMSE score, higher the prediction power)

We can thus conclude that our model can make good predictions about the prices of used cars.

We also performed cross-validation on the training part of the train-test split. We used the k-folds cross-validation, with k selected to be 30 (as the training set had a lot of observations, a lower k value would take more time to be processed).

Below are the results:

1. R-Square scores: (Thirty values will be generated)

```
array([ 7.34785957e-01,  8.06142546e-01,  8.01797865e-01,  7.97178086e-01,
        7.91405628e-01,  7.65001065e-01, -1.02231654e+22,  7.71869312e-01,
        8.04566268e-01,  6.61548396e-01,  8.02967580e-01,  7.91073419e-01,
        7.89590647e-01,  7.79145692e-01,  8.36124531e-01,  6.53862040e-01,
        8.11863767e-01,  8.51715645e-01,  7.85833309e-01,  7.13596936e-01,
        6.37200903e-01,  7.86209712e-01,  7.67271521e-01,  7.06065857e-01,
        8.30534138e-01,  8.11020636e-01,  8.20808635e-01,  7.89431684e-01,
        8.07658935e-01,  8.13373209e-01])
```

```
Mean of r2 scores: -3.407721801469782e+20
Standard Deviation of r2 scores: 1.835114351778006e+21
Maximim of r2 scores: 0.8517156453758572
Minimum of r2 scores: -1.0223165404409347e+22
Median of r2 scores: 0.7903320329796902
```

Most of the r-square scores are high, i.e., closer to 1, which indicates that the model explains most of the variance in the data. (except for one outlier value of `-1.022e+22`)

2. RMSE values: (Thirty values will be generated)

```
array([5.05132808e+00, 6.01982015e+00, 5.40905250e+00, 5.41136029e+00,
       4.50130831e+00, 5.30337837e+00, 1.12488932e+12, 5.84432632e+00,
       4.31193970e+00, 7.88084170e+00, 5.06952789e+00, 4.49334468e+00,
       5.98743035e+00, 5.07708498e+00, 4.21689929e+00, 5.95330807e+00,
       3.83915728e+00, 3.73703076e+00, 4.02245277e+00, 4.18209957e+00,
       9.05236312e+00, 5.03688366e+00, 5.29974228e+00, 6.05220650e+00,
       4.81713476e+00, 4.42348175e+00, 4.41828905e+00, 6.07082309e+00,
       5.20237155e+00, 4.79348226e+00])
```

```
Mean of RMSE values: 37496310830.94918
Standard Deviation of RMSE values: 201923813456.07498
Maximum of RMSE values: 1124889324776.9968
Minimum of RMSE values: 3.737030755400465
Median of RMSE values: 5.073306433021662
```

Most of the RMSE values are low (between 3 and 10 lakh INR) (except for the one outlier value), which means that the model has good prediction power.

The higher mean, std, and maxima of the RMSE values is due to one very high RMSE value present in the array (1.12 * 10 ^ 12)