USED CAR PRICE PREDICTION USING LINEAR REGRESSION

Guide : mr.vivek

Team members :

M.karthikeyan-20140105

M.Mohammed Sadham Hussain-20150106

G.Ranganathan-201408

Introduction:

The used car market is a vast and complex one with many variables that determine the price of a vehicle. Some of the factors that determine the price of a used car include age, mileage, condition, make and model, location, and market demand. In this project, we will explore the factors that determine the price of a used car and build a predictive model that can estimate the price of a used car.

Data Collection:

The first step in our project is to collect data on used cars. We can obtain this data from various sources, including online marketplaces, classified ads, and dealership websites. We need to collect data on various variables that can affect the price of a used car, including make and model, year of manufacture,

mileage, condition, and location. Once we have collected the data, we need to clean and preprocess it to ensure that it is accurate and consistent.

Data Analysis:

The next step is to analyze the data and identify the variables that have the most significant impact on the price of a used car. We can use various statistical techniques and machine learning algorithms to analyze the data and identify the variables that are most predictive of the price of a used car. Some of the techniques we can use include regression analysis, correlation analysis, and decision trees.

Model Building:

Once we have identified the variables that are most predictive of the price of a used car, we can build a predictive model. We can use various machine learning algorithms to build the model, including linear regression, decision trees, and neural networks. We will need to split the data into training and testing sets to evaluate the performance of the

model.

Model Evaluation:

The final step in our project is to evaluate the performance of the model. We can use various metrics to evaluate the performance of the model, including mean absolute error, mean squared error, and R-squared. We need to ensure that the model is accurate and consistent in predicting the price of a used car.

Conclusion:

In conclusion, the used car market is a complex one with many variables that determine the price of a vehicle. By collecting and analyzing data on used cars, we can build a predictive model that can estimate the price of a used car. The model can be useful for individuals looking to buy or sell a used car and for businesses involved in the used car market.

DATA COLLECTION

- The data collection process for a used car price project typically involves collecting data on various variables that can affect the price of a used car. This data can be collected from various sources, including online marketplaces, classified ads, and dealership websites.

- To collect the data, one would typically perform web scraping or use APIs to extract the relevant information from these sources. The extracted data would then be stored in a structured format, such as a CSV file or a database, for further analysis.

- It is important to ensure that the data collected is accurate and consistent. This may involve

cleaning and preprocessing the data to remove any inconsistencies, such as misspelled names or inconsistent formatting. It may also involve verifying the data against external sources, such as manufacturer specifications or third-party valuation tools.

- Overall, the data collection process is a critical step in a used car price project, as it lays the foundation for the subsequent data analysis and modeling.

In [2]:
```
df = pd.read_csv('/kaggle/input/used-cars-price-prediction/train-data.csv')
df.head()
```

Out[2]:

	Unnamed: 0	Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type
0	0	Maruti Wagon R LXI CNG	Mumbai	2010	72000	CNG	Manual	First
1	1	Hyundai Creta 1.6 CRDi SX Option	Pune	2015	41000	Diesel	Manual	First
2	2	Honda Jazz V	Chennai	2011	46000	Petrol	Manual	First
3	3	Maruti Ertiga VDI	Chennai	2012	87000	Diesel	Manual	First
4	4	Audi A4 New 2.0 TDI Multitronic	Coimbatore	2013	40670	Diesel	Automatic	Second

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6019 entries, 0 to 6018
Data columns (total 14 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   Unnamed: 0         6019 non-null   int64  
 1   Name               6019 non-null   object 
 2   Location           6019 non-null   object 
 3   Year               6019 non-null   int64  
 4   Kilometers_Driven  6019 non-null   int64  
 5   Fuel_Type          6019 non-null   object 
 6   Transmission       6019 non-null   object 
 7   Owner_Type         6019 non-null   object 
 8   Mileage            6017 non-null   object 
 9   Engine             5983 non-null   object 
 10  Power              5983 non-null   object 
 11  Seats              5977 non-null   float64
 12  New_Price          824 non-null    object 
 13  Price              6019 non-null   float64
```

DATA ANALYSIS

- Data analysis is the process of examining and interpreting data using various analytical and statistical techniques to derive meaningful insights and inform decision-making. The goal of data analysis is to identify patterns, trends, and relationships within the data that can be used to answer specific questions, make predictions, and draw conclusions.

- The process of data analysis typically involves the following steps:

- Define the problem: The first step in data analysis is to define the problem or question you want to answer with your data.

- Collect the data: Once you have defined your problem, you need to collect the relevant data. This may involve gathering data from various sources, such as surveys, interviews, or online databases.

- Clean and preprocess the data: Raw data often contains errors, missing values, or outliers that can skew the results of your analysis. Cleaning and preprocessing the data involves removing or correcting errors, filling in missing values, and transforming the data into a format that can be easily analyzed.

- Analyze the data: Once the data has been cleaned and preprocessed, you can begin to analyze it using

various statistical and analytical techniques. This may involve visualizing the data using charts and graphs, performing descriptive statistics to summarize the data, or running more complex statistical tests to identify patterns and relationships within the data.

- Interpret the results: After analyzing the data, you need to interpret the results and draw conclusions. This involves identifying key findings, determining their significance, and drawing insights that can inform decision-making.

- Communicate the results: The final step in data analysis is to communicate your findings to stakeholders. This may involve creating visualizations, presenting your results in a report or presentation, or using the data to make recommendations or inform decision-making.

- Overall, data analysis is a powerful tool that can help individuals and organizations make better decisions, improve their operations, and gain a deeper understanding of the world around them.

Data Cleaning

In [4]:
```
round((df.isna().sum()/len(df))*100,2)
```

Out[4]:
```
Unnamed: 0           0.00
Name                 0.00
Location             0.00
Year                 0.00
Kilometers_Driven    0.00
Fuel_Type            0.00
Transmission         0.00
Owner_Type           0.00
Mileage              0.03
Engine               0.60
Power                0.60
Seats                0.70
New_Price           86.31
Price                0.00
dtype: float64
```

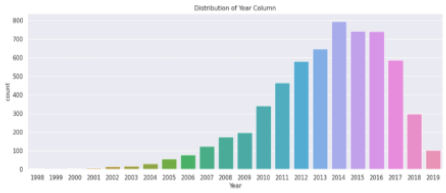
Univariate Analysis

In [17]:
```
def remove_outliers(df,col,lower,upper):
    df = df[ (df[col]>lower) & (df[col]<upper) ]
    return df
```

In [18]:
```
def plot_num(df,col):
    fig ,ax = plt.subplots(1,2,figsize=(16,4))
    sns.histplot(df,x=col,kde=True,ax=ax[0])
    sns.boxplot(df,x=col,ax=ax[1])
    ax[0].set_title(f'Distribution of {col}')
    ax[1].set_title(f'{col} Boxplot')
    fig.show();
```

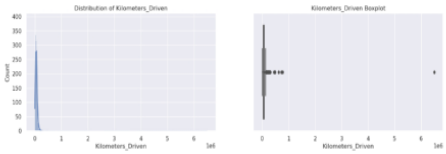
1. Year

In [19]:
```
plt.figure(figsize=(15,5))
sns.countplot(df,x='Year');
plt.title('Distribution of Year Column');
```



2. Kilometers Driven

In [20]:
```
plot_num(df,'Kilometers_Driven')
```

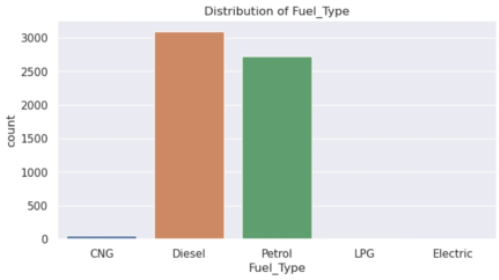


In [21]:
```
df = remove_outliers(df,'Kilometers_Driven',0,150000)
```

In [22]:
```
plot_num(df,'Kilometers_Driven')
```

3. Fuel Type

In [23]:
```
plt.figure(figsize=(8,4))
sns.countplot(df,x='Fuel_Type');
plt.title('Distribution of Fuel_Type');
```

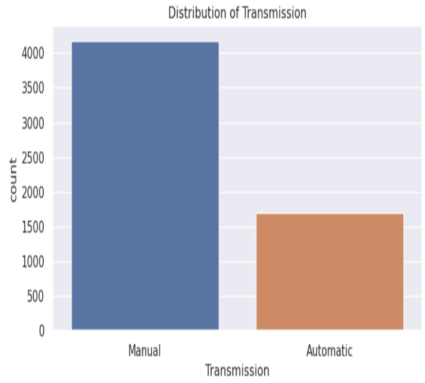


In [24]:
```
df['Fuel_Type'].value_counts()
```

Out[24]:
```
Diesel      3098
Petrol      2726
CNG           56
LPG           10
Electric       2
Name: Fuel_Type, dtype: int64
```

4. Transmission

In [27]:
```
plt.figure(figsize=(8,4))
sns.countplot(df,x='Transmission');
plt.title('Distribution of Transmission');
```

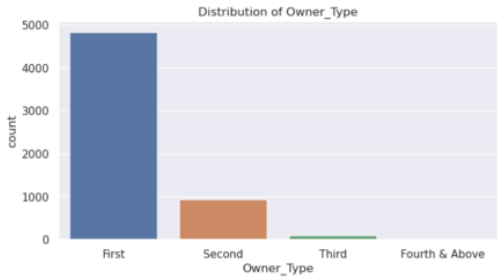


In [28]:
```
df['Transmission'].value_counts()
```

Out[28]:
```
Manual       4176
Automatic    1704
Name: Transmission, dtype: int64
```

5. Owner Type

In [29]:
```
plt.figure(figsize=(8,4))
sns.countplot(df,x='Owner_Type');
plt.title('Distribution of Owner_Type');
```

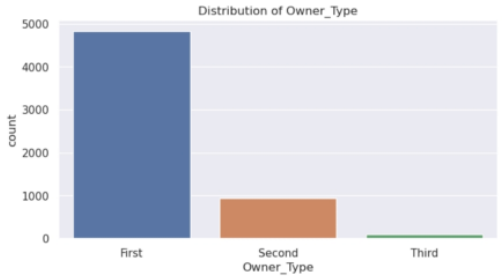


In [30]:
```
df['Owner_Type'].value_counts()
```

Out[30]:
```
First              4835
Second              934
Third               105
Fourth & Above        6
Name: Owner_Type, dtype: int64
```

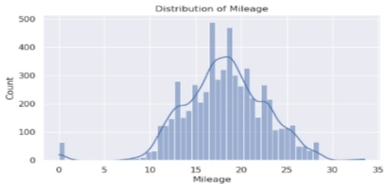
In [31]:
```
df = df[df.Owner_Type!='Fourth & Above']
```

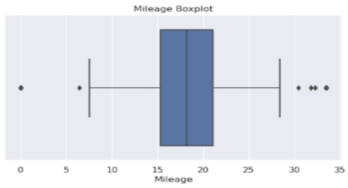
In [32]:
```
plt.figure(figsize=(8,4))
sns.countplot(df,x='Owner_Type');
plt.title('Distribution of Owner_Type');
```



6. Mileage

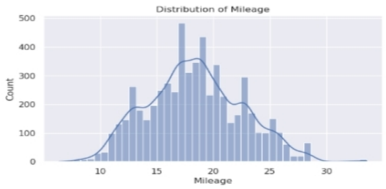
In [33]:
```
plot_num(df,'Mileage')
```

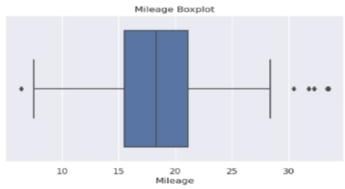




In [34]:
```
df = remove_outliers(df,'Mileage',0,35)
```

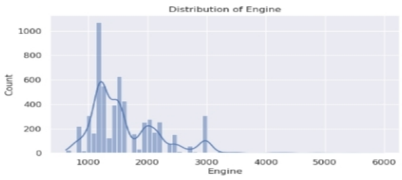
In [35]:
```
plot_num(df,'Mileage')
```

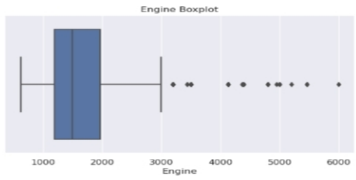




7. Engine

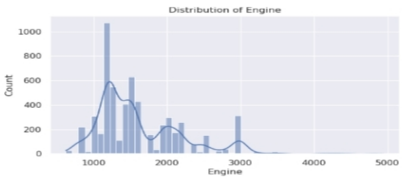
In [36]:
```
plot_num(df,'Engine')
```

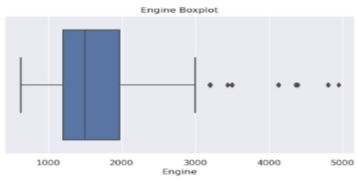




In [37]:
```
df = remove_outliers(df,'Engine',0,5000)
```

In [38]:
```
plot_num(df,'Engine')
```





Model building:

To build a used car price prediction model, you should follow these steps:

- Collect data on used car prices from various sources
- Clean and preprocess the data to remove errors and missing values
- Create new features that may be useful for the model
- Select a suitable model such as linear regression, decision trees, random forests or neural networks
- Train the model on your dataset and evaluate its performance
- Fine-tune the model by adjusting its hyperparameters
- Deploy the model and make it accessible to your audience.
- By following these steps, you can build a model that accurately predicts the prices of used cars

Building Model

In [69]:

```
from sklearn.model_selection import train_test_split,RandomizedSearchCV,GridSear
chCV ,cross_val_score
from sklearn.pipeline import make_pipeline
from sklearn.linear_model import LinearRegression,Ridge ,Lasso
from sklearn.metrics import mean_squared_error,r2_score
from sklearn.preprocessing import StandardScaler,RobustScaler , OneHotEncoder,Po
lynomialFeatures ,PowerTransformer
from sklearn.impute import SimpleImputer
from sklearn.compose import make_column_transformer
from category_encoders import BinaryEncoder
```

Linear Regression model :

Linear regression is a widely used statistical method for modeling the relationship between a dependent variable and one or more independent variables. After building a linear regression model, it is important to evaluate its performance to determine how well it fits the data and whether it is suitable for making predictions.

Here are some common methods for evaluating a linear regression model:

- Residual analysis: Residuals are the differences between the actual values of the dependent variable and the predicted values from the model. A good linear regression model should have small residuals that are randomly distributed around zero. Plotting the residuals against the predicted values can help identify patterns or trends that may indicate a problem with the model.

- R-squared: R-squared is a measure of how much of the variation in the dependent variable is explained by the independent variables in the model. An R-squared value of 1 indicates that the model perfectly fits the data, while a value of 0 indicates that the model explains none of the variation in the data. Generally, a higher R-squared value indicates a better fit, but it should be used in conjunction with other measures of model performance.

- Root Mean Squared Error (RMSE): RMSE is a measure of the difference between the actual values of the dependent variable and the predicted values from the model. It is calculated by taking the square root of the average of the squared differences between the predicted and actual values. A lower RMSE indicates a better fit.

- Adjusted R-squared: Adjusted R-squared is a modified version of R-squared that takes into account the number of independent variables in the model. It penalizes models with a large number of independent variables that do not significantly contribute to the fit of the model.
- Cross-validation: Cross-validation is a method of evaluating the performance of a model by splitting the data into training and testing sets. The model is trained on the training set and then tested on the testing set. This process is repeated multiple times to get a more robust estimate of the model's performance. Cross-validation can help identify overfitting, which occurs when the model fits the training data too closely and does not generalize well to new data

LinearRegression

In [73]:
```
lin_reg = make_pipeline(
            preprocessor,
            LinearRegression()
    )
lin_reg.fit(X_train,y_train)
```

Out[73]:
```
Pipeline(steps=[('columntransformer',
                 ColumnTransformer(transformers=[('pipeline-1',
                                                  Pipeline(steps=[('simpleimp
uter',
                                                                   SimpleImpu
ter()),
                                                                  ('standards
caler',
                                                                   StandardSc
aler()),
                                                                  ('polynomia
lfeatures',
                                                                   Polynomial
Features(degree=3))]),
                                                  Index(['Year', 'Kilometers_
Driven', 'Mileage', 'Engine', 'Power', 'Seats'], dtype='object')),
                                                 ('pipeline-2',
                                                  Pipeline(steps=[('simpleimp
uter',
                                                                   SimpleImpu
ter(strategy='most_frequent')),
                                                                  ('onehotenc
oder',
                                                                   OneHotEnco
der())]),
                                                  Index(['Location', 'Fuel_Ty
pe', 'Transmission', 'Owner_Type', 'Company',
       'Model'],
      dtype='object'))])),
                ('linearregression', LinearRegression())])
```