A PROJECT REPORT

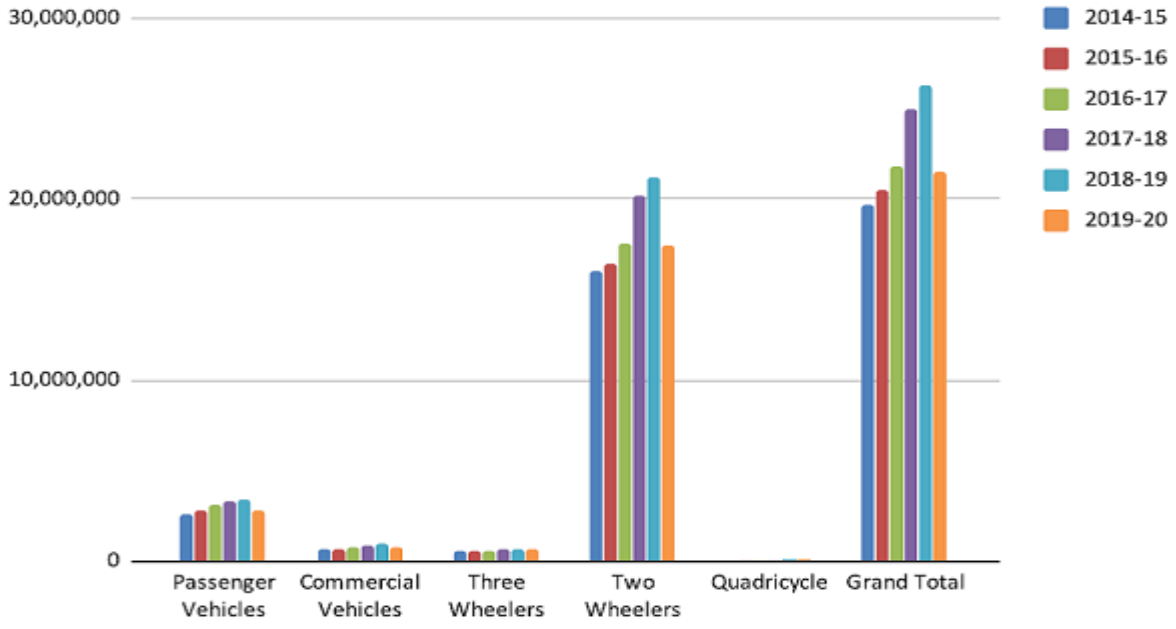
ON

USED BIKES PRICES IN INDIA

INTRODUCTION

The increase in the number of two wheelers in India has been considerably high in the past 15 years. The two wheelers are targeted by the vast middle class population because it did not make dent on the pockets. It started with scooters but slowly the markets evolved and we got LML, Honda and many new players entered in the fray and the 2 wheelers became extremely popular because of their affordability and availability .

If you see the situation presently, in total vehicles sales, the share of two-wheelers is the most as per data released by SIAM(The Society of Indian Manufacturers)



With the increase in number of bike owners, one of the possible solutions can be buying used bikes instead of new ones. This would fulfil the needs of people and also not increase the number of bikes in the country. Selling a used bike would depend on lots of components like brand, age, horsepower, kilometres travelled and its working condition. This report would help us to study such fronts and use machine learning techniques to predict the approximate selling/buying value of these used bikes.

Problem statement

The aim to model a resale valuation for used bikes and predict the price of used bikes. This can be helpful while selling a used bike or buying a used bike.

This dataset contains information about approx. 32000 used bikes scraped from www.droom.in .This dataset comprises bikes a range of all used bikes sold on droom.in. It includes features like power, kilometres drive, Age of the bike etc.

Concepts Used

Data analysis –

Data analysis is a process of inspecting, cleansing, transforming, and modelling data with the goal of discovering useful information, informing conclusions, and supporting decision-making. Data analysis has multiple facets and approaches, encompassing diverse techniques under a variety of names, and is used in different business, science, and social science domains. In today's business world, data analysis plays a role in making decisions more scientific and helping businesses operate more effectively.

Data mining is a particular data analysis technique that focuses on statistical modelling and knowledge discovery for predictive rather than purely descriptive purposes, while business intelligence covers data analysis that relies heavily on aggregation, focusing mainly on business information. In statistical applications, data analysis can be divided into descriptive statistics, exploratory data analysis (EDA), and confirmatory data analysis (CDA). EDA focuses on discovering new features in the data while CDA focuses on confirming or falsifying existing hypotheses. Predictive analytics focuses on the application of statistical models for predictive forecasting or classification, while text analytics applies statistical, linguistic, and structural techniques to extract and classify information from textual sources, a species of unstructured data. All of the above are varieties of data analysis.

Data Visualization-

Data visualization is the representation of data through use of common graphics, such as charts, plots, infographics, and even animations. These visual displays of information communicate complex data relationships and data-driven insights in a way that is easy to understand.

Data visualization can be utilized for a variety of purposes, and it's important to note that is not only reserved for use by data teams. Management also leverages it to convey organizational structure and hierarchy while data analysts and data scientists use it to discover and explain patterns and trends.

Types of data visualizations

The earliest form of data visualization can be traced back the Egyptians in the pre-17th century, largely used to assist in navigation. As time progressed, people leveraged data visualizations for broader applications, such as in economic, social, health disciplines. Perhaps most notably, Edward Tufte published The Visual Display of Quantitative Information, which illustrated that individuals could utilize data visualization to present data in a more effective manner. His book continues to stand the test of time, especially as companies turn to dashboards to report their performance metrics in real-time. Dashboards are effective data visualization tools for tracking and visualizing data from multiple data sources, providing visibility into the effects of specific behaviours by a team or an adjacent one on performance. Dashboards include common visualization techniques, such as:

- **Tables**: This consists of rows and columns used to compare variables. Tables can show a great deal of information in a structured way, but they can also overwhelm users that are simply looking for high-level trends.
- **Pie charts and stacked bar charts:** These graphs are divided into sections that represent parts of a whole. They provide a simple way to organize data and compare the size of each component to one other.
- **Line charts and area charts:** These visuals show change in one or more quantities by plotting a series of data points over time and are frequently used within predictive analytics. Line graphs utilize lines to demonstrate these changes while area charts connect data points with line segments, stacking variables on top of one another and using colour to distinguish between variables.
- **Histograms**: This graph plots a distribution of numbers using a bar chart (with no spaces between the bars), representing the quantity of data that falls within a particular range. This visual makes it easy for an end user to identify outliers within a given dataset.
- **Scatter plots**: These visuals are beneficial in revealing the relationship between two variables, and they are commonly used within regression data analysis. However, these can sometimes be confused with bubble charts, which are used to visualize three variables via the x-axis, the y-axis, and the size of the bubble.
- **Heat maps:** These graphical representation displays are helpful in visualizing behavioural data by location. This can be a location on a map, or even a webpage.

- **Tree maps**, which display hierarchical data as a set of nested shapes, typically rectangles. Tree maps are great for comparing the proportions between categories via their area size.

Machine Learning

Machine learning (**ML**) is a field of inquiry devoted to understanding and building methods that 'learn', that is, methods that leverage data to improve performance on some set of tasks.[1] It is seen as a part of artificial intelligence. Machine learning algorithms build a model based on sample data, known as training data, in order to make predictions or decisions without being explicitly programmed to do so. Machine learning algorithms are used in a wide variety of applications, such as in medicine, email filtering, speech recognition, and computer vision, where it is difficult or unfeasible to develop conventional algorithms to perform the needed tasks.

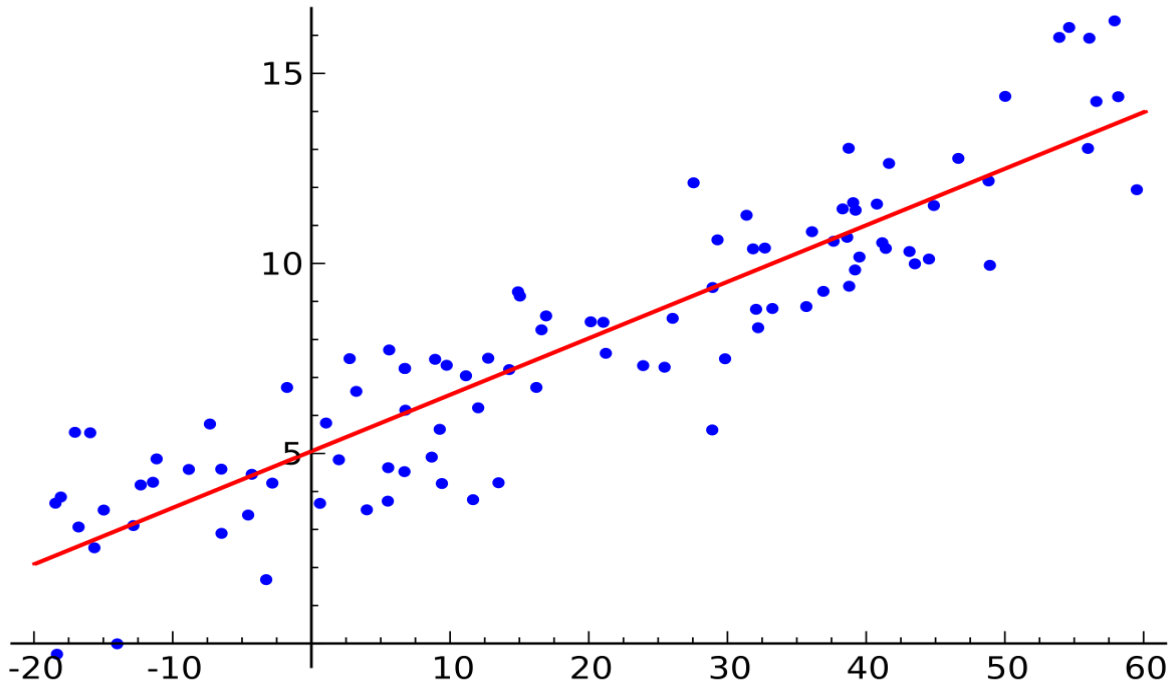
A subset of machine learning is closely related to computational statistics, which focuses on making predictions using computers, but not all machine learning is statistical learning. The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning. Data mining is a related field of study, focusing on exploratory data analysis through unsupervised learning. Some implementations of machine learning use data and neural networks in a way that mimics the working of a biological brain. In its application across business problems, machine learning is also referred to as predictive analytics.

In the following model we will be using Linear regression and Decision Tree Regression.

Linear Regression

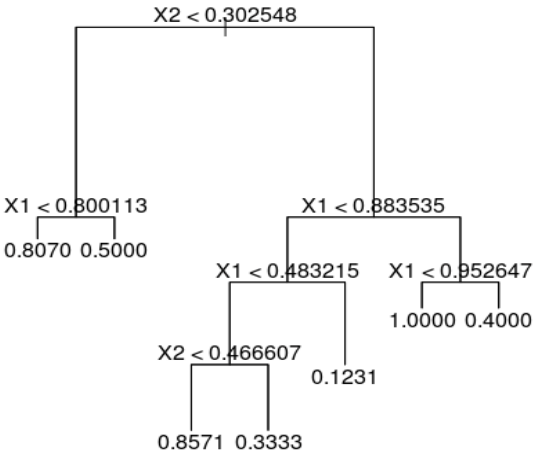
Linear regression is one of the easiest and most popular Machine Learning algorithms. It is a statistical method that is used for predictive analysis. Linear regression makes predictions for continuous/real or numeric variables such as **sales, salary, age, product price,** etc.

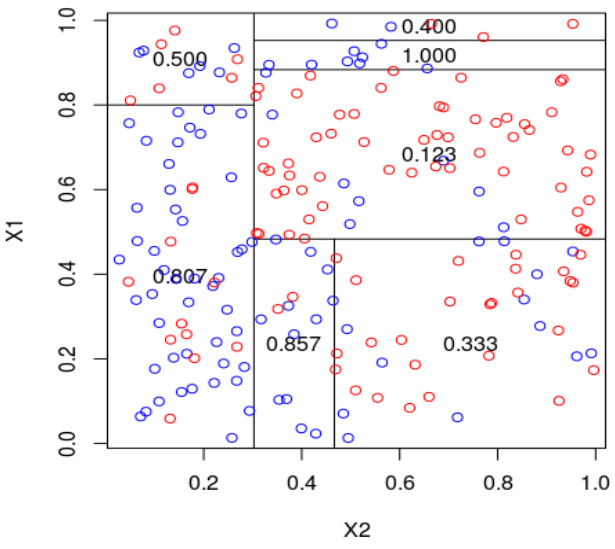
Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (y) variables, hence called as linear regression. Since linear regression shows the linear relationship, which means it finds how the value of the dependent variable is changing according to the value of the independent variable.



Decision Tree Regression

Decision tree builds regression or classification models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with **decision nodes** and **leaf nodes**. A decision node (e.g., Outlook) has two or more branches (e.g., Sunny, Overcast and Rainy), each representing values for the attribute tested. Leaf node (e.g., Hours Played) represents a decision on the numerical target. The topmost decision node in a tree which corresponds to the best predictor called **root node**. Decision trees can handle both categorical and numerical data.





METHODOLOGY

IMPORTING NECESSARY LIBRARIES AND READUNG THE DATAFRAME

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import *
from sklearn.linear_model import *
from sklearn import metrics
%matplotlib inline
import warnings
warnings.filterwarnings("ignore")
```

Next we explore data and use data analysis techniques using the above libraries.

```
bikes.head()
```

	bike_name	price	city	kms_driven	owner	age	power	brand
0	TVS Star City Plus Dual Tone 110cc	35000.0	Ahmedabad	17654.0	First Owner	3.0	110.0	TVS
1	Royal Enfield Classic 350cc	119900.0	Delhi	11000.0	First Owner	4.0	350.0	Royal Enfield
2	Triumph Daytona 675R	600000.0	Delhi	110.0	First Owner	8.0	675.0	Triumph
3	TVS Apache RTR 180cc	65000.0	Bangalore	16329.0	First Owner	4.0	180.0	TVS
4	Yamaha FZ S V 2.0 150cc-Ltd. Edition	80000.0	Bangalore	10000.0	First Owner	3.0	150.0	Yamaha

```
bikes.shape
```

```
(32648, 8)
```

The given dataset contains 8 features(columns) and 32648 entries(rows) which might contain some of the duplicate entries

```
#dropping the duplicate entries
bikes=bikes.drop_duplicates()
```

```
bikes.shape
#rechecking the shape after dropping duplicate values
```

```
(7324, 8)
```

We see that after dropping duplicate values , the number of entries reduce from 32,648 to 7324. Our next step is to extract information from data using pandas.

```
#sorting values in descending order
bikes['price'].sort_values(ascending=False )
```

```
367    1900000.0
4864   1800000.0
7671   1550000.0
6872   1500000.0
3016   1500000.0
       ...
4923      6700.0
4922      6400.0
1382      5800.0
9194      5000.0
4960      4400.0
Name: price, Length: 7324, dtype: float64
```

The maximum price of bike is 19,00,000(19 lakhs) and minimum price is 4,400

```
#Names of bike who are old and yet expensive(more than 10 lakhs
bikes.loc[(bikes['price']>100000 )& (bikes['age']>15.0)][['bike_name','age','price']]
```

	bike_name	age	price
334	Royal Enfield Bullet 350 cc	39.0	125000.0
1278	Royal Enfield Bullet 350cc	18.0	130000.0
1976	Royal Enfield Standard 350cc	27.0	250000.0
2980	Royal Enfield Bullet 350cc	43.0	135000.0
5216	Royal Enfield Bullet 350cc	37.0	140000.0
5866	Royal Enfield Bullet 350cc	39.0	115000.0
6685	Royal Enfield Bullet 350cc	38.0	120000.0

These are some of the vintage bikes as they are more than 15 years old and yet they are expensive

```
#counting number of bikes of different brands
bikes['brand'].value_counts()
```

```
Bajaj              2081
Royal Enfield      1346
Hero               1142
Honda               676
Yamaha              651
TVS                 481
KTM                 375
Suzuki              203
Harley-Davidson      91
Kawasaki             61
Hyosung              53
Mahindra             50
Benelli              46
Triumph              21
Ducati               20
BMW                  10
Jawa                  7
Indian                3
MV                    3
Rajdoot               1
LML                   1
Yezdi                 1
Ideal                 1
Name: brand, dtype: int64
```

Bajaj is the most common bike in india followed by Royal Enfield Hero and Honda

```
#number of bikes in a particular city
bikes['city'].value_counts().head(10)
```

```
Delhi        1426
Bangalore     683
Mumbai        609
Gurgaon       474
Faridabad     463
Pune          394
Jaipur        299
Chennai       288
Ahmedabad     249
Ghaziabad     242
Name: city, dtype: int64
```

It seems Delhi has the maximum number of bike owners followed by Bangalore and Mumbai.

These are some of the most populous cities in INDIA which explains more number of owners

```
#expensive bikes count city wise
bikes[bikes['price']>1000000]['city'].value_counts()
```

```
Delhi        8
Mumbai       3
Pune         2
Chennai      1
Jaipur       1
Alibag       1
Ludhiana     1
Chandrapur   1
Gurgaon      1
Ghaziabad    1
Bangalore    1
Hyderabad    1
Hubli        1
Name: city, dtype: int64
```

Delhi has 8 bikes costing more than 10 lakhs followed by Mumbai Pune

It tells us that Delhi has more people who are financialy sound as they can afford bikes costing more than 10,00,000

```
#counting city with most old bikes
bikes[bikes['age']>10]['city'].value_counts().head(10)
```

```
Delhi        150
Bangalore     69
Gurgaon       67
Ahmedabad     54
Mumbai        53
Pune          52
Chennai       46
Faridabad     42
Ghaziabad     37
Noida         26
Name: city, dtype: int64
```

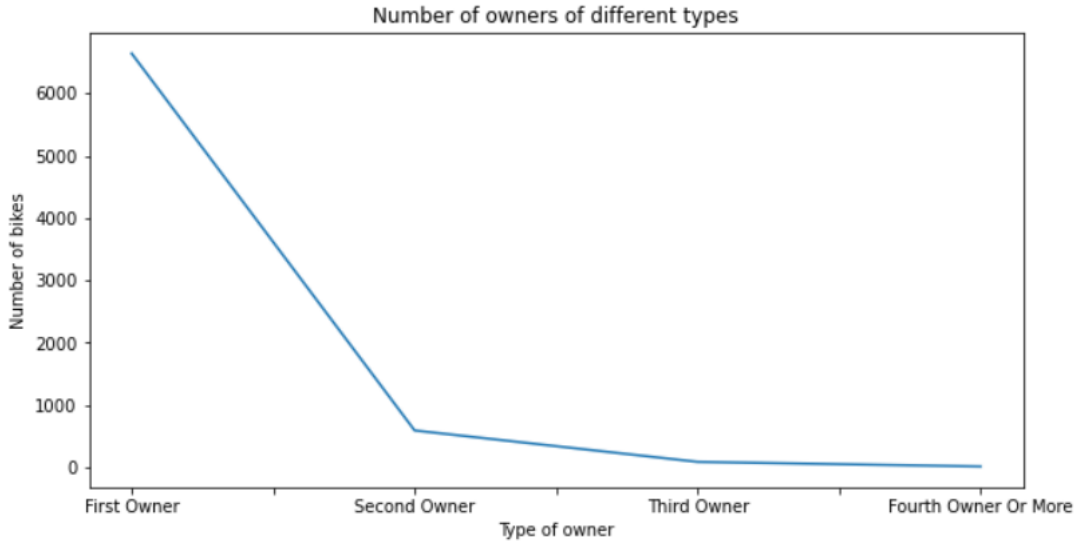
People in above cities like to keep their bikes for a longer duration of time (especially DELHI)

```
#city wise count of bikes with power more than 1000
bikes[bikes['power']>1000][['power','city']].groupby('city').count()
```

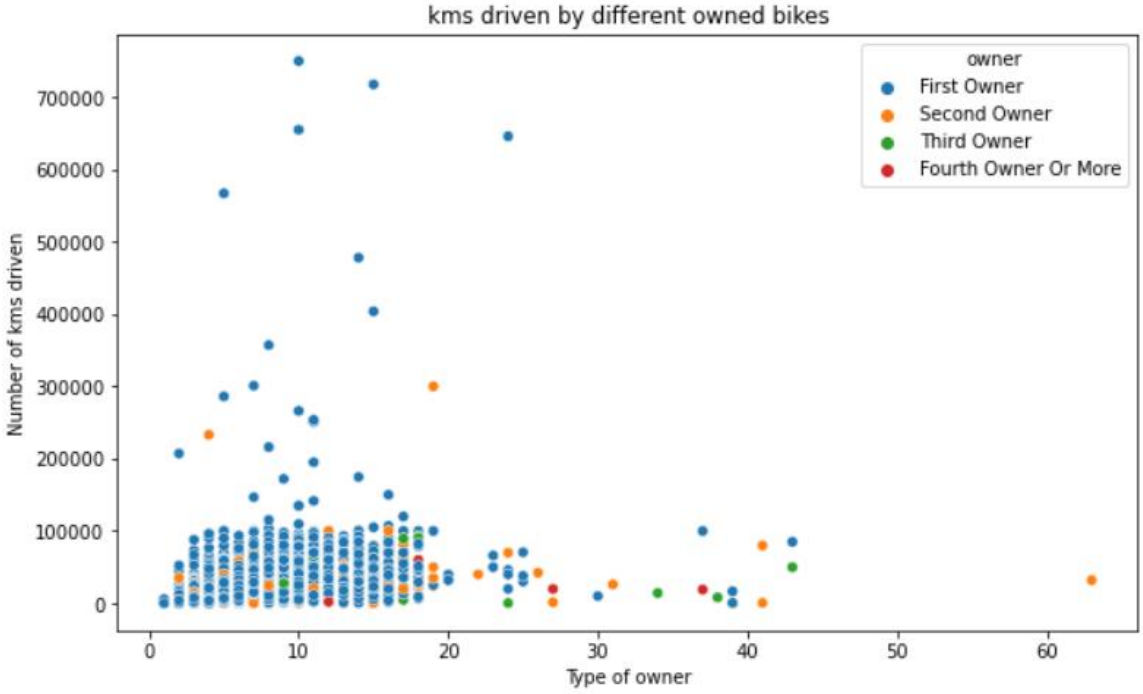
city	power
Ahmedabad	2
Alibag	1
Bangalore	2
Chandrapur	1
Chennai	2
Coimbatore	2
Delhi	7
Ghaziabad	1
Gurgaon	1
Guwahati	1
Hyderabad	2
Jaipur	1
Koppal	1
Mumbai	3
Nagpur	2
Tiruvallur	1

```
#scaling price of bikes in hundred for better visualization
aa=bikes['price']/100
bikes['pricein100']=aa
bikes.drop('price',axis=1,inplace=True)
```

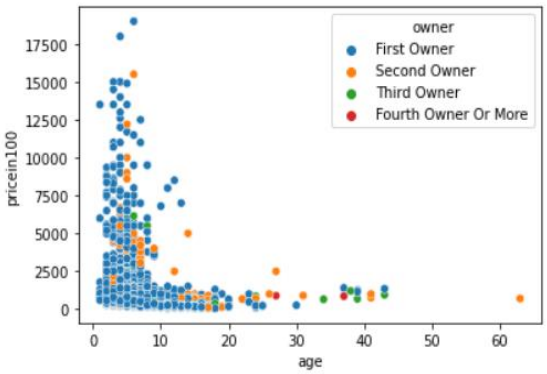
In the next step we will be using matplotlib and seaborn libraries to visualize our dataset and take out the valuable information regarding our dataset which will be useful in training our machine learning model.

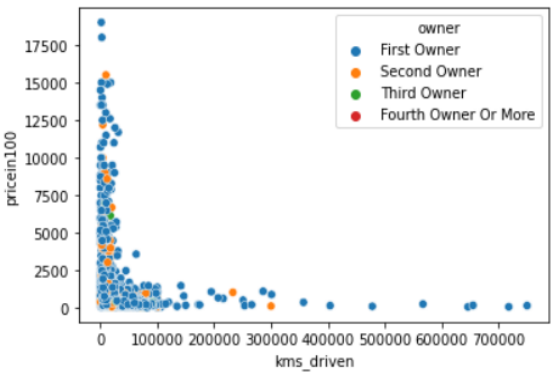


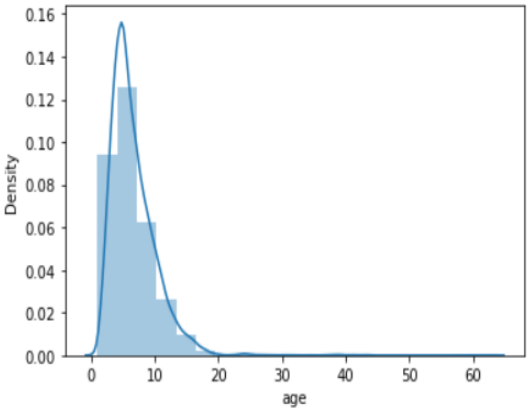
The above plot tells that most of the bikes are owned by first owner and almost negligible people are third and fourth owner

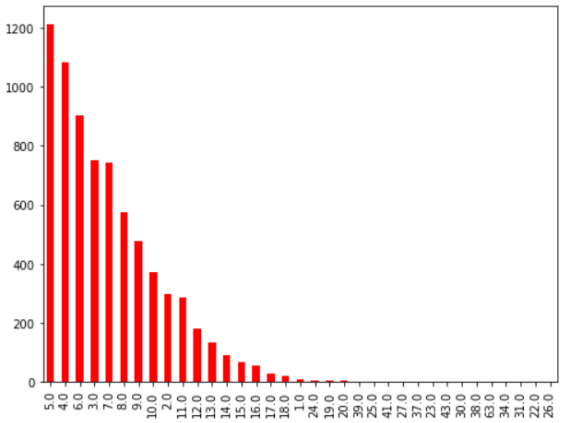


Maximum number of kms driven are by the first owners and other owners bike haven't travelled much.





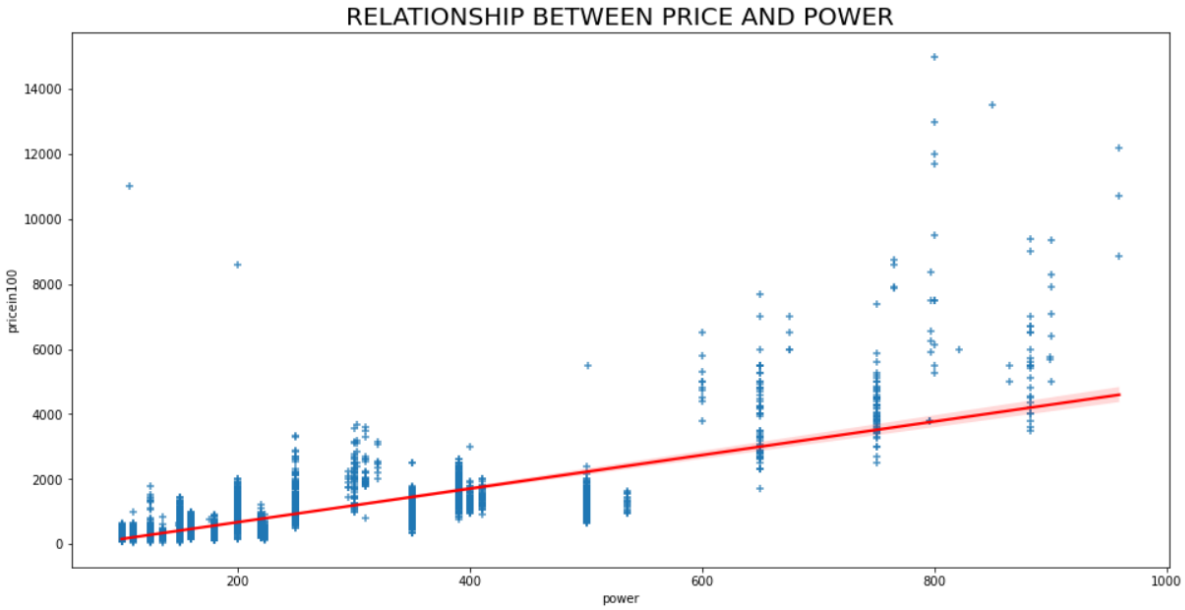




Above plot indicates most of the bikes are 3-10 years old which is the normal duration a bike lasts with an owner in INDIA

We analyse that most expensive bikes are the new ones and their price decreases almost exponentially with age.

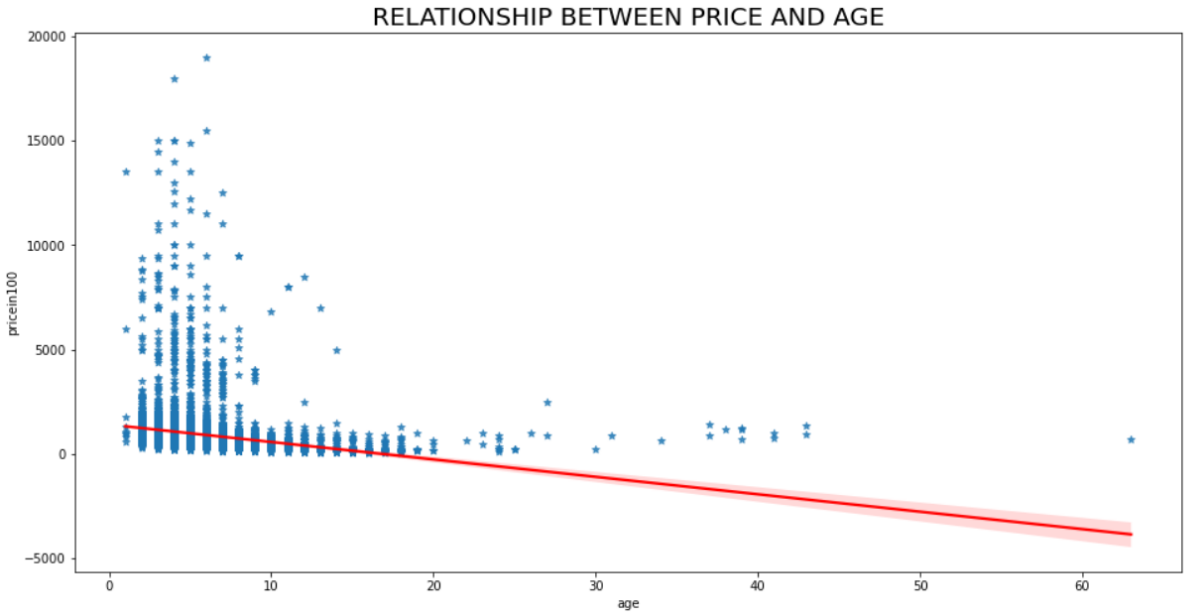
```
#plotting power vs price with restricting power to 1000 to avoid outliers
ss=bikes[bikes['power']<1000]
plt.figure(figsize=[16,8])
sns.regplot(ss['power'],ss['pricein100'],line_kws={"color": "red"},marker='+')
plt.title("RELATIONSHIP BETWEEN PRICE AND POWER",fontdict={'color':'black','fontsize':20})
```



1.Most bike range in power of 100-650 and price upto 2 lakhs

2.There seems to be a linear relation between power and price

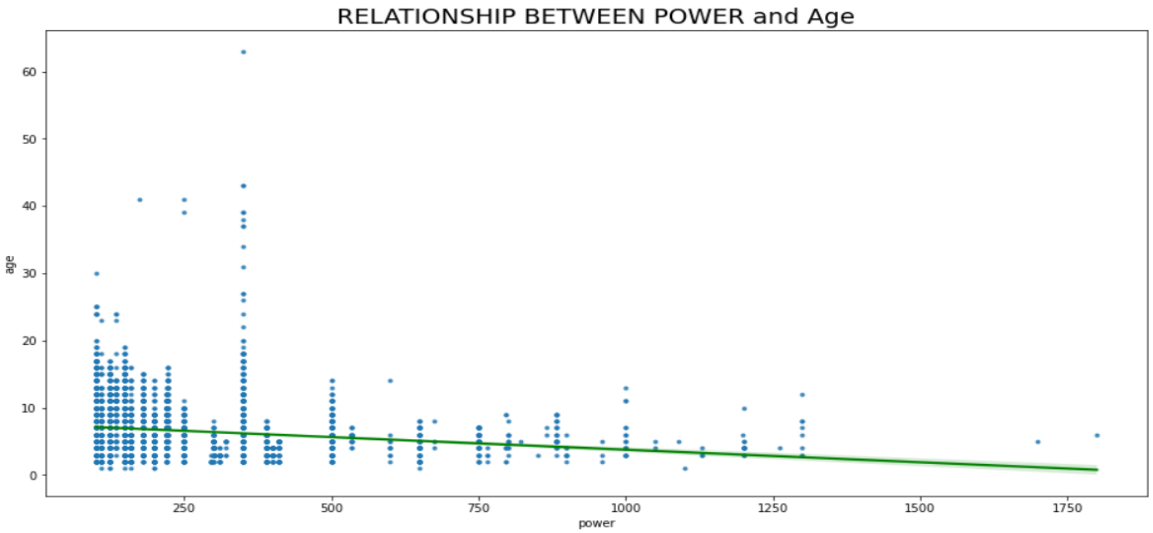
3.Power and Price are directly proportional to each other

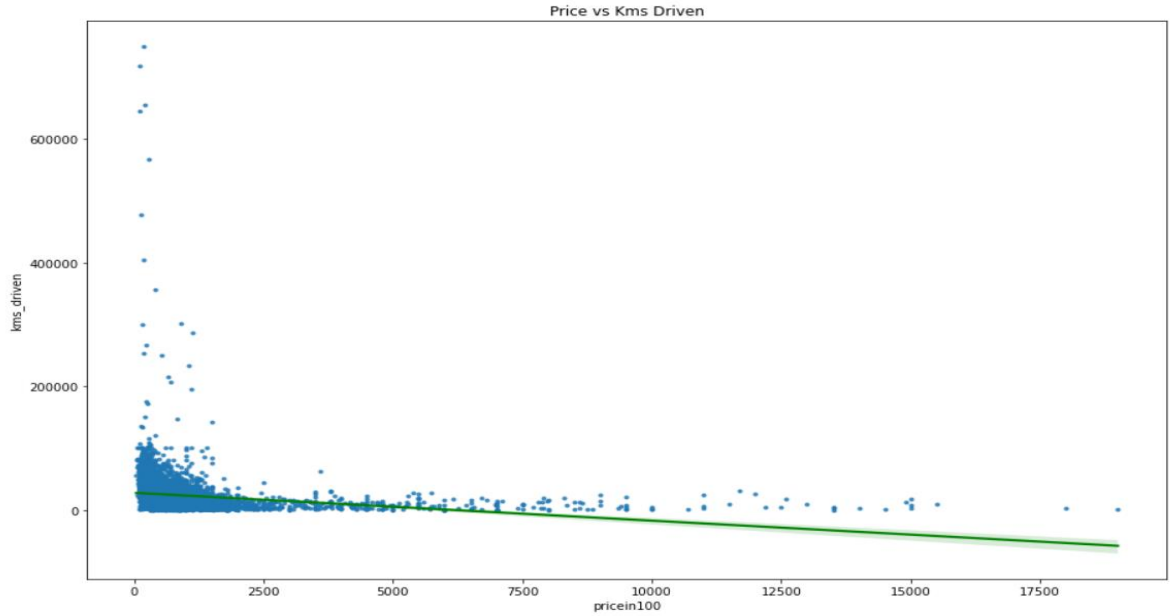


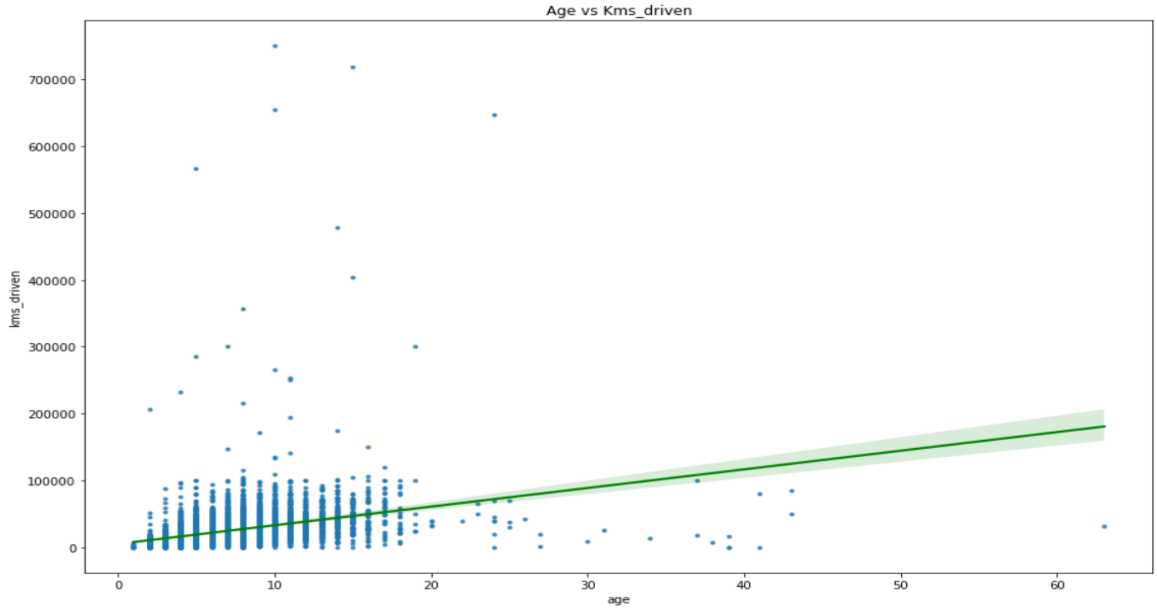
1. Most of the bikes are less than 10 years old.

2. With incrase in age the price of bike decreases which indicate inverse relation between each other.

3. The graph seems to have too many outliers(vintage bikes and sports bikes)


RELATIONSHIP BETWEEN POWER and Age
age
power
0
10
20
30
40
50
60
250
500
750
1000
1250
1500
1750


Price vs Kms Driven
kms_driven
pricein100
0
200000
400000
600000
0
2500
5000
7500
10000
12500
15000
17500


Age vs Kms_driven
kms_driven
age
0
100000
200000
300000
400000
500000
600000
700000
0
10
20
30
40
50
60

1.Old bikes have travelled less kms than new ones because they were not so efficient and long lasting.

2.Bikes that are 5-10 year old have travelled most as they are consistent and pocket friendly

3.Bikes with less power have travelled more and number of kms driven changes with increase in power.

4.Bikes with less power are less fuel consuming and so they are most commonly used while more power bikes are used in races or for a collection so they travel less.

5.With increase in price number of kms travelled reduces.

6.Cheap bikes are mostly used to travel by common people in india . Its their means of transport which is the reason why they have travelled more.

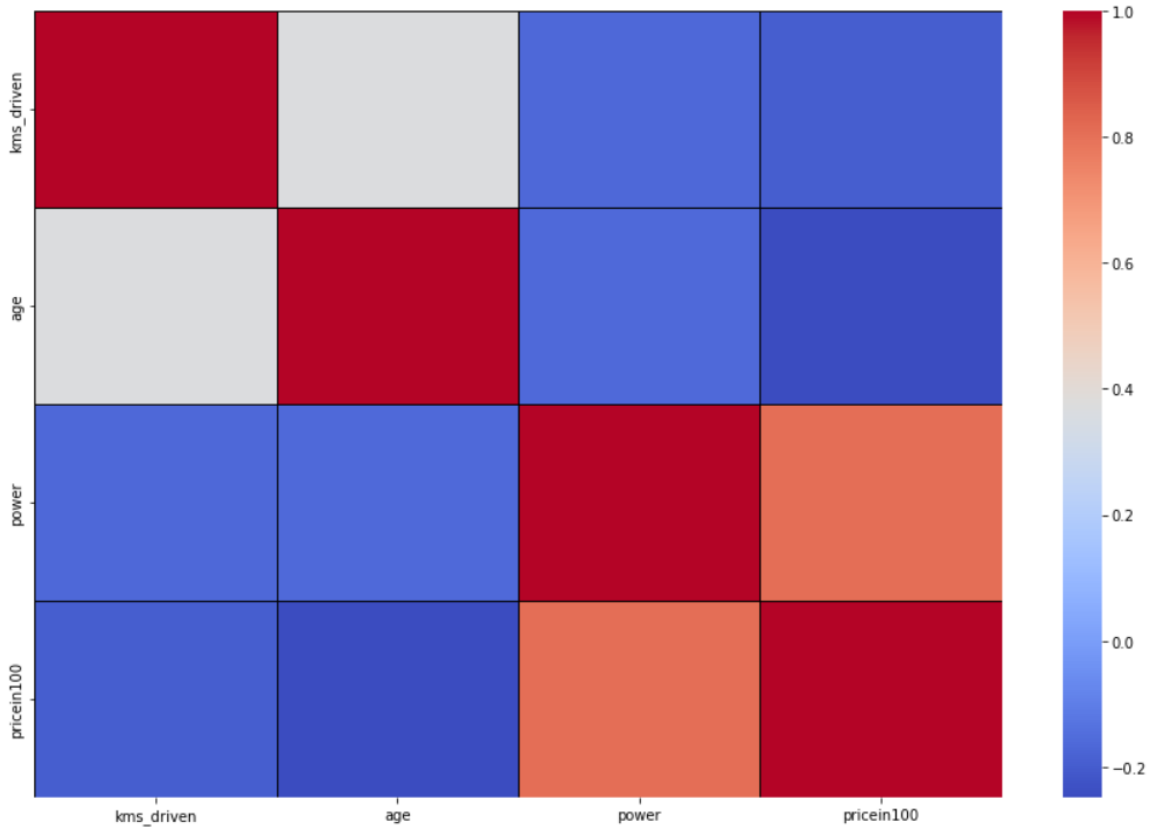
7.Expensive/Costly bikes are mostly used either for races or as a collection which is the reason they travel less.

8.With increase in age we see reduction in power.

9.This can be interpreted to the fact that old bikes have low power engines and with time new bikes with more power were invented.

```
#heatmap
plt.figure(figsize=(15,10))
sns.heatmap(corri,linewidths=1,linecolor='black',cmap='coolwarm')
```

```
<AxesSubplot:>
```



After analysing and visualising the dataset, we next will start training the model with steps. The first step will be data preprocessing in which we will define the independent and dependent variables, clean data of outliers, use feature scaling if necessary (which was not necessary in this case), using one hot encoding to convert categorical data to Numerical data.

Now what is CATEGORICAL DATA ?

Categorical data are variables that contain n label values rather than numeric values. The number of possible values is often limited to a fixed set. Categorical variables are often called nominal.

What is the problem with categorical data?

Many machine learning algorithms cannot operate on label data directly. They require all input variables and output variables to be numeric.

In general, this is mostly a constraint of the efficient implementation of machine learning algorithms rather than hard limitations on the algorithms themselves.

This means that categorical data must be converted to a numerical form. If the categorical variable is an output variable, you may also want to convert predictions by the model back into a categorical form in order to present them or use them in some application.

How to Convert Categorical Data to Numerical Data?

This involves two steps:

1. Integer Encoding
2. One-Hot Encoding

1. Integer Encoding

As a first step, each unique category value is assigned an integer value.

For example, “*red*” is 1, “*green*” is 2, and “*blue*” is 3.

This is called a label encoding or an integer encoding and is easily reversible.For some variables, this may be enough.

The integer values have a natural ordered relationship between each other and machine learning algorithms may be able to understand and harness this relationship.

2. One-Hot Encoding

For categorical variables where no such ordinal relationship exists, the integer encoding is not enough.

In fact, using this encoding and allowing the model to assume a natural ordering between categories may result in poor performance or unexpected results (predictions halfway between categories).

In this case, a one-hot encoding can be applied to the integer representation. This is where the integer encoded variable is removed and a new binary variable is added for each unique integer value.

In the "*color*" variable example, there are 3 categories and therefore 3 binary variables are needed. A "1" value is placed in the binary variable for the color and "0" values for the other colors.

Code below is one of the way to do One HOT Encoding which we will be using for our model to encode brand and owner columns.

```
#Getting dummies for brands and owners
owner_dummies=pd.get_dummies(bikes.owner)
owner_dummies.head()
brand_dummies=pd.get_dummies(bikes.brand)
brand_dummies.head()
```

```
merged_bike = pd.concat([bikes,owner_dummies,brand_dummies],axis=1)
merged_bike.head()
```

```
merged_bike.shape
```

```
(7324, 35)
```

	bike_name	city	kms_driven	owner	age	power	brand	pricein100	First Owner	Fourth Owner Or More	...	LML	MV	Mahindra	Rajdoot	Royal Enfield	Suzuki	TVS	Triumph	Ya
0	TVS Star City Plus Dual Tone 110cc	Ahmedabad	17654.0	First Owner	3.0	110.0	TVS	350.0	1	0	...	0	0	0	0	0	0	1	0	
1	Royal Enfield Classic 350cc	Delhi	11000.0	First Owner	4.0	350.0	Royal Enfield	1199.0	1	0	...	0	0	0	0	1	0	0	0	
2	Triumph Daytona 675R	Delhi	110.0	First Owner	8.0	675.0	Triumph	6000.0	1	0	...	0	0	0	0	0	0	0	1	
3	TVS Apache RTR 180cc	Bangalore	16329.0	First Owner	4.0	180.0	TVS	650.0	1	0	...	0	0	0	0	0	0	1	0	
4	Yamaha FZ S V 2.0 150cc-Ltd. Edition	Bangalore	10000.0	First Owner	3.0	150.0	Yamaha	800.0	1	0	...	0	0	0	0	0	0	0	0	

Next we drop the unnecessary columns like owner, city, brand,fourth owner or more as they either have been encoded to numerical datatype or they are not correlated to the price of bikes.

```
#dropping unnecessary columns
merged_bike.drop(['brand','bike_name','city','owner','Fourth Owner Or More'],axis=1,inplace=True)
```

The code below is used to take care of the outliers and since our dataset is so large removing some of the values won't have much effect on our Machine Learning model.

```
#removing the outliers to filter a proper dataset
merged_bike=merged_bike[merged_bike['price']<10000]
merged_bike=merged_bike[merged_bike['age']<15]
merged_bike=merged_bike[merged_bike['power']<1200]
merged_bike=merged_bike[merged_bike['kms_driven']<100000]
```

Our next step is to split our dataset to training and testing dataset.

Why do we do that?

The train-test split procedure is used to estimate the performance of machine learning algorithms when they are used to make predictions on data not used to train the model.

It is a fast and easy procedure to perform, the results of which allow you to compare the performance of machine learning algorithms for your predictive modeling problem. Although simple to use and interpret, there are times when the procedure should not be used, such as when you have a small dataset and situations where additional configuration is required, such as when it is used for classification and the dataset is not balanced.

Train-Test Split Procedure in Scikit-Learn

The scikit-learn Python machine learning library provides an implementation of the train-test split evaluation procedure via the train_test_split() function.
The function takes a loaded dataset as input and returns the dataset split into two subsets.

```
#Independent variables are atored in x and dependent feature(price) stored in y
x=merged_bike.iloc[:,:-1]
y=merged_bike.iloc[:,-1]
```

```
#Splitting into training and testing dataset
x_train, x_test, y_train, y_test= train_test_split(x, y, test_size= 0.2,random_state=5)
```

Test size- It describes the size of testing dataset which in this case is 20% of our dataframe.

Random State- Different types of randomization tasks are involved in machine learning models and other related functions. When splitting a dataset, splitting a node in a decision tree or a random forest, initializing centroids in clustering, randomization takes place. The random state hyperparameter is used to control any such randomness involved in machine learning models to get consistent results.

Training our dataset is followed by defining a model and fitting it in the training dataset.

In Scikit-learn we have inbuild models available to fit our data. Presently we will use Linear Regression.

```
#Defining the regression model
regression = LinearRegression()
regression.fit(x_train,y_train)
print('Training Complete')
```

```
#testing dataset after defining a model
pred=regression.predict(x_test)
pred
```

```
#finding out the accuracy of our regression model
accuracy=(regression.score(x_test,y_test))*100
print('Model Accuracy:', accuracy)
```

```
Model Accuracy: 90.04351862154513
```

```
# Finding MAE and MSE
print('Mean Absolute Error:' ,
      metrics.mean_absolute_error(y_test,pred))
print('Mean Squared Error:',
      metrics.mean_squared_error(y_test, pred))
```

```
Mean Absolute Error: 179.2418462018757
Mean Squared Error: 81670.58566336584
```

```
# r2 score
from sklearn.metrics import r2_score
r2_score(y_test,pred)
```

```
0.9004351862154513
```

```
df= pd.DataFrame({'RealData':y_test, 'PredictedData':pred})
df
```

	RealData	PredictedData
2259	1200.0	1266.786255
2490	350.0	538.388561
4244	190.0	-78.287696
6379	1140.0	1613.947617
6373	420.0	495.564883
...	...	...
1553	800.0	678.622766
3276	600.0	559.345618
1861	900.0	638.234975
8606	2504.3	2044.135442
2361	1320.0	1060.414080

1408 rows × 2 columns

Findings

1.The accuracy of our linear regression model is 90%.

2.The model performs well.

3. The MSE is 81670.58 and MAE is 179.241 which is not much but can be further reduced.

There were certain assumptions we made while using our linear regression model(Although we saw the linearity between the features during data visualization) .

Assumptions of simple linear regression

Simple linear regression is a parametric test, meaning that it makes certain assumptions about the data. These assumptions are:

1. Homogeneity of variance (homoscedasticity): the size of the error in our prediction doesn't change significantly across the values of the independent variable.
2. Independence of observations: the observations in the dataset were collected using statistically valid sampling methods, and there are no hidden relationships among observations.
3. Normality: The data follows a normal distribution.

4. The relationship between the independent and dependent variable is linear: the line of best fit through the data points is a straight line (rather than a curve or some sort of grouping factor).

USING DECISION TREE ALGORITHM for our MODEL.

As to improve our model, next we try using Decision Tree Regressor to train and test our model.

```
#defining Decision tree model
from sklearn.tree import DecisionTreeRegressor
dt=DecisionTreeRegressor()
```

```
x_train, x_test, y_train, y_test= train_test_split(x, y, test_size= 0.2,random_state=5)
```

```
dt.fit(x_train,y_train)
print('Training Complete')
```

```
pred=dt.predict(x_test)
pred
```

```
array([1250.,  650.,  230., ...,  750., 2125., 1100.])
```

```
accuracy=(dt.score(x_test,y_test))*100
print('Model Accuracy:', accuracy,"%")
print(r2_score(y_test,pred))
```

```
Model Accuracy: 93.36241605751093 %
0.9336241605751092
```

```
print('Mean Absolute Error:' ,
      metrics.mean_absolute_error(y_test,pred))
print('Mean Squared Error:',
      metrics.mean_squared_error(y_test, pred))
```

```
Mean Absolute Error: 145.30567471590908
Mean Squared Error: 54446.480374672516
```

```
df= pd.DataFrame({'RealData':y_test, 'PredictedData':pred})
df
```

	RealData	PredictedData
2259	1200.0	1250.0
2490	350.0	650.0
4244	190.0	230.0
6379	1140.0	1020.0
6373	420.0	426.3
...	...	...
1553	800.0	800.0
3276	600.0	550.0
1861	900.0	750.0
8606	2504.3	2125.0
2361	1320.0	1100.0

1408 rows × 2 columns

Findings

1. The accuracy of the model is 93.36%
2. Model performs better than the linear regression model.
3. The MSE is 54446.48 and MAE is 145.30 which is a good improvement in comparison to previous model.

Summary

1. Our aim to predict price of used bikes is completed.
2. Data has been properly analysed and visualized.
3. India is a populous and developing country which is the reason why people prefer bikes more and that too with less cost.
3. Expensive bikes are rarely bought in the country and if bought are rarely used to travel.
4. People with financial issues mostly pass their bikes to the next generation and use it until it lasts and is in working condition.
5. Indians prefer repairing than buying a new one.
6. Our Decision tree model performs better than the Linear Regression maybe because of non-linearity between some of the variables.

Conclusion-

Our model worked better with decision trees than with linear regression. Other options like Feature Scaling, Feature Engineering and Feature Transformation(using Ensemble and Boosting Algorithms like gradient boosting, Hyperparameter tuning) were used while training our model but they seem to reduce the accuracy of the model and hence were removed in the final run. The training was looped to find the perfect random state to get best accuracy and hence the value five was chosen. After removing extreme values or Outliers about 2.5% of the data was lost which can be reduced when studied more cautiously and which might result in better results.

Other Regression model were also applied but Decision tree seemed to trump them all. There can be many other approaches which would have lead to different or maybe better outcomes. There's a lot of story left to be told by this dataset and my aim will be to continue working on it. The takings from this model will be that every DATA has its own uniqueness and conclusions , it's the right tools and algorithms that help in breaking it. Further in this deep learning techniques like Neural Networks can be applied. Everyone has his/her own perspective and its upto you how you read and understand it.