Project Report – Used Cars Price Prediction

Submitted by:

Sahil Kumar (Internship – 24)

Acknowledgement

I'd like to acknowledge the invaluable assistance of the Data Trained Academy and FlipRobo Technologies teams in providing guidance to work on real-time data projects, which also assisted me in doing a lot of research and obtaining insights.

FlipRobo Technologies have provided all the necessary information.

The data was obtained through web scraping from cardekho.com and support was obtained from stackoverflow.com, medium.com, github.com, scikit-learn.org, towardsdatascience.com, analyticsvidhya.com and w3schools.com, during the data collection and model building phases of the project.

INTRODUCTION

Business Problem Framing

With the covid 19 impact in the market, we have seen lot of changes in the car market. Now some cars are in demand hence making them costly and some are not in demand hence cheaper. One of our clients works with small traders, who sell used cars. With the change in market due to covid 19 impact, our client is facing problems with their previous car price valuation machine learning models. So, they are looking for new machine learning models from new data. We must make a car price valuation model.

The main aim of this project is to predict the price of used car based on various features. Machine Learning is a field of technology developing with immense abilities and applications in automating tasks. So, we will deploy an ML model for car selling price prediction and analysis. This kind of system becomes handy for many people. This model will provide the approximate selling price for the car based on different features like fuel type, transmission, price, weight, running in kms, engine displacement, milage etc and this model will help the client to understand the price of used cars.

Conceptual Background of the Domain Problem

Car Price Prediction is a fascinating machine learning problem because there are numerous factors that influence the price of a used car on the market. It is common in many developed countries to lease a car rather than buy it outright. A lease is a legally binding contract between a buyer and a seller (or a third party – usually a bank, insurance company, or other financial institution) in which the buyer agrees to pay fixed monthly/yearly payments to the seller/financer.

When the lease period expires, the buyer has the option to purchase the vehicle at its residual value, which is its expected resale value. As a result, it is commercially advantageous for sellers/financiers to be able to accurately predict the salvage value (residual value) of cars. If the seller/financer initially underestimates the residual value, the instalments will be higher for the clients, who will almost certainly switch to another seller/financer. If the residual value is overestimated, the clients' monthly payments will be lower, but the seller/financer may have a difficult time selling these high-priced used cars at the overestimated residual value. As a result, we can see that estimating the price of used cars is also of great commercial importance.

By deploying machine learning models, we are attempting to assist the client who works with small traders who sell used cars in understanding the price of the used cars. These models would assist clients/sellers in understanding the used car market and, as a result, selling the used car in the market.

Review of Literature

The literature review examines relevant literature to gain insight into the factors that are important in predicting the market price of used cars. In this paper, we discuss various applications and methods that inspired us to develop our supervised ML techniques for predicting used car prices in various locations. We conducted a background survey on the fundamental ideas of our project and used those ideas to collect data information by web scraping from the www.cardekho.com website, which is a web platform where sellers can sell their used cars.

This project is more concerned with data exploration, feature engineering, and pre-processing that can be performed on this data. We can do better data exploration and derive some interesting features using the available columns because we scrape a large amount of data that includes more car-related features. To make the predictions, various techniques such as ensemble techniques, k-nearest neighbours, and decision trees were used.

The goal of this project is to create an application that can predict car prices using other features. In the long run, this would allow people in this increasingly digital world to better explain and review their purchases with one another.

Motivation for the Problem Undertaken

It can be tough to determine whether a used car is worth the advertised price while looking at ads online. Mileage, engine displacement, running, make, model, year, and other characteristics can all affect a car's true value. It might be difficult to price a second-hand car correctly from the standpoint of a seller.

The major goal is to develop models for predicting used automobile prices using machine learning algorithms.

- Create a supervised machine learning model for predicting a vehicle's worth based on several attributes.
- The system being developed must be feature-based, allowing for feature-wise prediction.
- Providing graphical comparisons for a clearer picture

ANALYTICAL PROBLEM FRAMING

Mathematical / Analytical Modelling of the Problem

When it comes to data science, we're using mathematical models to model business circumstances, the environment, and other factors, and we can gain more insights from these models, such as the outcomes of our decisions, what we should do next, and how we should do it to improve the odds. As a result, mathematical models are vital, and choosing the proper one to answer a business question can be extremely beneficial to a company.

We need to create a machine learning model that can estimate the price of used autos that is both efficient and effective. As a result, "Car Price" is our continuous target variable. It's clear that we're dealing with a regression problem, and we'll need to apply regression techniques to forecast the outcomes.

The model building required two phases as a part of this project:

1. Data Collection: I utilised web scraping to obtain data about used automobiles from the well-known website www.cardekho.com, where I found more car features than on other websites and was able to get data for several areas. As per our client's request, we must create a model to forecast the pricing of these old cars.
2. Model Building: I created a machine learning model after gathering the data. Have completed all data pre-processing processes prior to developing the model. The following is the whole data science life cycle that I employed in this project:
    a. Data cleaning
    b. EDA
    c. Data pre-processing
    d. Model building
    e. Model evaluation
    f. Selecting the best model
    g. Predicting prices using the best model

Data Sources and their Formats

The data was gathered from the website www.cardekho.com, which is a web platform where sellers can list their used cars for sale. The data is scraped using the Web scraping technique and the Selenium framework.

We scraped over 6,462 records and grabbed data for various cities across India, collecting information on various car attributes, and saving the information in excel format. The final dataset, post data cleaning has 6,224 rows and 11 columns, with the target variable "Price" being one of them. The dataset in question contains both categorical and numerical data. The following is the data description:

S. No.	Variable	Definition
1	Make_Year	Year of manufacture of the vehicle
2	Driven_Kilometers	Number of kilometres driven by the current owner as on the date of posting on website
3	Fuel	Type of fuel used in the car, i.e., Petrol, Diesel, CNG, etc.
4	Transmission	Type of gear transmission in the vehicle, i.e., automatic, manual etc.
5	Owner(s)	Number of previous owners as on the date of posting on the website
6	Mileage	Number of kilometres the vehicle can drive on 1 litre of fuel
7	Engine	Cubic capacity of the engine on the vehicle
8	Location	Location / place of registration of the vehicle at the time of posting on the website
9	Brand	Brand name / company which manufactured the vehicle
10	Price	Asking price of the vehicle, as posted on the website

Data Pre-Processing done

The process of turning raw data into a machine-readable format for use by Machine Learning models is known as data pre-processing. Data pre-processing is an important stage in Machine Learning because the quality of data and the usable information that can be gleaned from it has a direct impact on our model's capacity to learn; consequently, pre-processing our data before feeding it into our model is critical. I utilised the pre-processing techniques below:

- Checked for null values
- Analysed the data types of each variable and fixed the types wherever necessary for optimum model building
- Analysed the data distribution across each variable
- Established relationships between different variables in the dataset
- Established correlations between different variables in the dataset
- Checked for outliers and removed them using IQR method
- Checked for skewness and used logs, square roots, and cube roots to remove the skewness
- Separated Brand Name of vehicle and created a new variable
- Encoded the categorical data

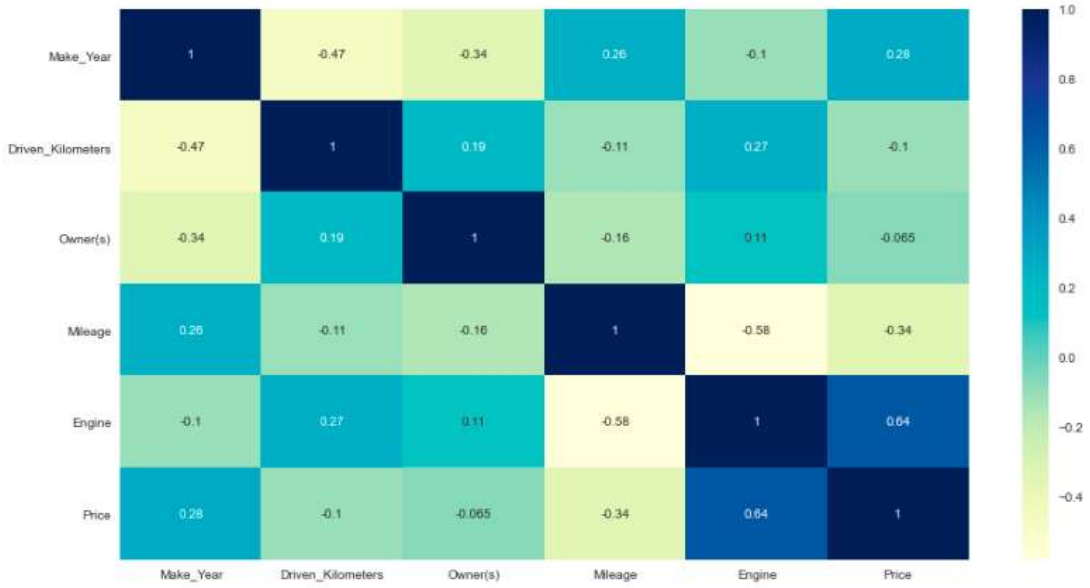
Data Input-Logic-Output Relationships

Labels and features make up the dataset. Because the values of our independent variables change as our label changes, the features are independent, and the label is dependent.

As a part of univariate analysis, I investigated the distribution of skewness using distplots for numerical features and count plots for categorical data because we had both numerical and categorical columns.

I utilised a variety of plotting approaches to investigate the relationship between features and the label using catplots, I discovered continuous variables that had a strong relationship with the label “Price”.

I checked the correlation between the numerical features and Price of the vehicle using a heatmap, which revealed both positive and negative correlations between the label and numerical features.



Hardware and Software Requirements & Tools Used

Hardware:

- Machine - Lenovo ThinkPad
- Processor - Intel(R) Core (TM) i5-8265U CPU @ 1.60GHz 1.80 GHz
- RAM – 8 GB
- System Type - 64-bit operating system, x64-based processor

Software:

- Windows 10 Pro
- MS Office 365
- Jupyter Notebook for coding in Python and building the machine learning model

Libraries imported and utilised:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline

from scipy.stats import zscore
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score,mean_squared_error,mean_absolute_error
from sklearn.ensemble import RandomForestRegressor
from sklearn.neighbors import KNeighborsRegressor
from sklearn.linear_model import Lasso,Ridge
from sklearn.tree import DecisionTreeRegressor
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import GridSearchCV

import warnings
warnings.filterwarnings('ignore')
```

- **NumPy** is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays
- **Pandas** is a software library written for the Python programming language for data manipulation and analysis. It offers data structures and operations for manipulating numerical tables and time series.
- **Seaborn** is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.
- **Matplotlib** is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits.
- **Scipy.stats** contains several probability distributions, summary and frequency statistics, correlation functions and statistical tests, masked statistics, kernel density estimation, quasi-Monte Carlo functionality, and more
- **Scikit-learn** (also known as sklearn) is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support-vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy. Scikit-learn is a NumFOCUS fiscally sponsored project.
- **Joblib** is a set of tools to provide lightweight pipelining in Python. Transparent disk-caching of functions and lazy re-evaluation (memorize pattern) and easy simple parallel computing. Joblib is optimized to be fast and robust on large data and has specific optimizations for NumPy arrays.

NumPy and Pandas libraries were used to process the dataset, seaborn and matplotlib were used in the visualization of data, Scipy.stats was used for attempting to remove outliers using z-score, skearn was used in pre-processing the dataset (encoding categorical data and standard scaling), creating a train test split, computing the metrics like MAE, MSE and R2 score, importing relevant regression algorithms for building the model, computing the cross validation scores, and hyperparameter tuning. Joblib was used exclusively for saving the model built on the training dataset and reloading the model for predicting prices of vehicles on the dataset.

MODEL/S DEVELOPMENT AND EVALUATION

Identification of possible problem-solving approaches

To tackle the problem, I employed both statistical and analytical methods, which included pre-processing the data and using EDA techniques and a heat map to examine the correlation of independent and dependent features. Removed outliers using the IQR method and used the log, square root, and cube root function to reduce the skewness. Encoded the categorical data using Label Encoder and Get Dummies techniques.

I made sure that the input data was cleaned and scaled before feeding it into the machine learning models. Looked for the optimal random state for the Regression model. Finally, evaluation metrics and numerous regression models were created and tested for their efficacy.

We needed to forecast the price of used cars for this project. The target variable in this dataset is Car Price, which means our target column is continuous, making this a regression problem. I forecasted the car price using a variety of regression models. After a series of tests, I determined that Random Forests is the best approach for creating our final model since it has a high R^2 score and lowest difference R^2 score and cross-validation score of all the algorithms tested.

I used K-Fold cross validation with cv=5 and then hyper parameter tuning on the best model to obtain good performance and evaluate whether my model was getting over-fitting and under-fitting. Then I saved and loaded my final model for predictions.

Testing of Identified Approaches

After pre-processing and data cleaning, I was left with 10 columns, including the target variable, and I used these independent features for model development and prediction. The following are the algorithms that were utilised to train the data:

- Random Forest Regressor
- K-Neighbors Regressor
- Decision Tree Regressor
- Lasso
- Ridge

Run and Evaluate Selected Models

We tested on 5 regression algorithms after choosing the random state with range 1 to 300 and used linear regression to find the best random state.

```
maxR2_Score = 0
maxRS = 0

for i in range(300):
    x_train,x_test,y_train,y_test = train_test_split(X_scaled,Y,test_size = 0.20,random_state = i)
    LR = LinearRegression()
    LR.fit(x_train,y_train)
    predrf = LR.predict(x_test)
    Score = r2_score(y_test,predrf)
    if Score>maxR2_Score:
        maxR2_Score = Score
        maxRS = i

print('The best accuracy is ',maxR2_Score, ' with Random State ',maxRS)
```

```
The best accuracy is  0.7853337991089029  with Random State  148
```

Model Building:

- 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.

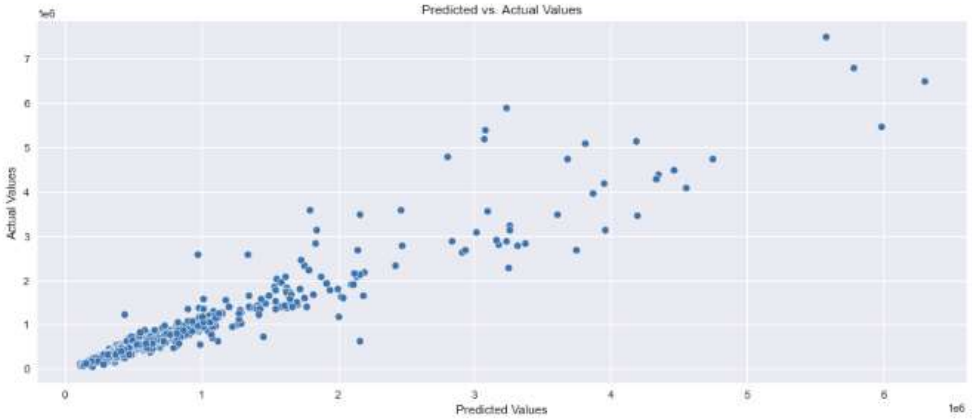
- Random Forest Regression is a supervised learning algorithm that uses ensemble learning method for regression. Ensemble learning method is a technique that combines predictions from multiple machine learning algorithms to make a more accurate prediction than a single model

- Decision tree regression observes features of an object and trains a model in the structure of a tree to predict data in the future to produce meaningful continuous output. Continuous output means that the output/result is not discrete, i.e., it is not represented just by a discrete, known set of numbers or values.

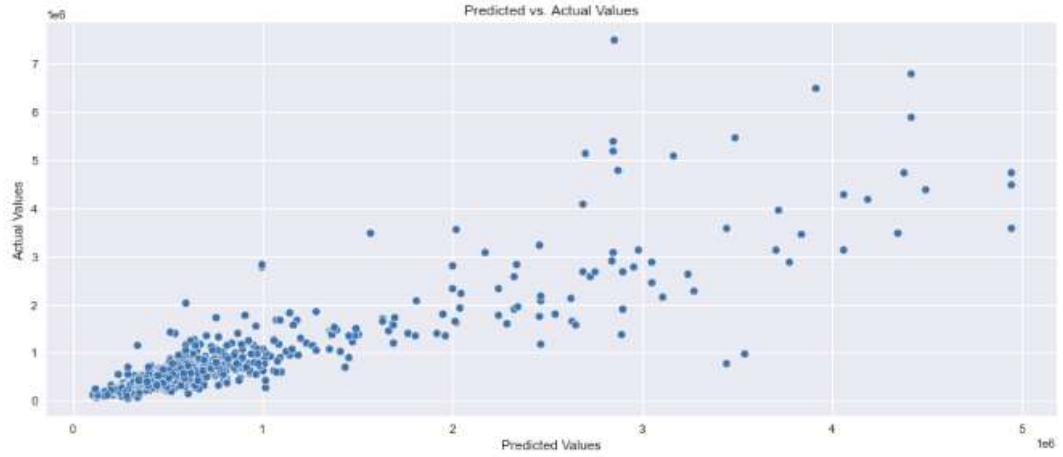
- The K nearest neighbors is a simple algorithm that stores all available cases and predict the numerical target based on a similarity measure and it uses ‘feature similarity’ to predict the values of any new data points. This means that the new point is assigned a value based on how closely it resembles the points in the training set.
- Lasso is a regression analysis method that performs both variable selection and regularization to enhance the prediction accuracy and interpretability of the resulting statistical model.
- Ridge regression is a method of estimating the coefficients of multiple-regression models in scenarios where linearly independent variables are highly correlated. It has been used in many fields including econometrics, chemistry, and engineering.

```
rf=RandomForestRegressor()
kn=KNeighborsRegressor()
dt = DecisionTreeRegressor(max_features='auto')
ls=Lasso()
rd=Ridge()

model=[rf,kn,dt,ls,rd]
kf = KFold(n_splits=5, random_state=43, shuffle=True)

train=[]
test=[]
cv=[]

for m in model:
    m = m.fit(x_train,y_train)
    pred_train=m.predict(x_train)
    pred_test=m.predict(x_test)
    train_score=r2_score(y_train,pred_train)
    train.append(train_score*100)
    test_score=r2_score(y_test,pred_test)
    test.append(test_score*100)
    print(m)
    print('R Squared (R2): ',test_score*100)
    print('Mean Squared Error (MSE): ',mean_squared_error(y_test,pred_test))
    print('Root Mean Squared Error (RMSE): ',np.sqrt(mean_squared_error(y_test, pred_test)))
    print('Mean Absolute Error (MAE): ',mean_absolute_error(y_test,pred_test))
    score=cross_val_score(m,X_scaled,Y,cv=kf)
    cv.append(score.mean()*100)
    plt.figure(figsize=[15,6])
    sns.scatterplot(x=pred_test, y=y_test)
    plt.xlabel('Predicted Values')
    plt.ylabel('Actual Values')
    plt.title('Predicted vs. Actual Values')
    plt.show()
    print('-'*120)



Overall_score={'Model':['RandomForest','KNN','DecisionTree Regressor','Lasso','Ridge'], 'Training Score':train, 'Test Score':tes

Overall_score=pd.DataFrame(data=Overall_score)
```

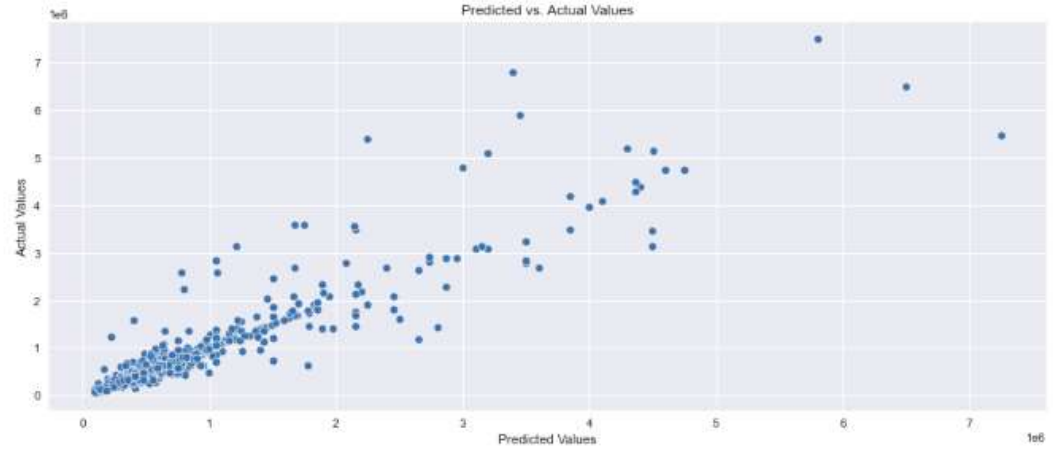
```
RandomForestRegressor()
R Squared (R2):  90.88377672361042
Mean Squared Error (MSE):  50950497108.42879
Root Mean Squared Error (RMSE):  225722.16795970392
Mean Absolute Error (MAE):  82787.13565045223
```



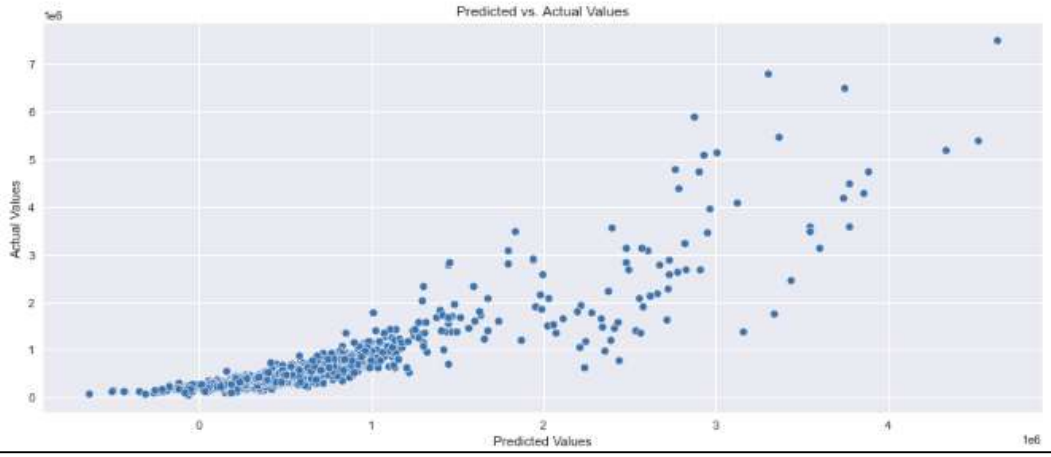
```
KNeighborsRegressor()
R Squared (R2):  77.65874769809187
Mean Squared Error (MSE):  124865075842.8838
Root Mean Squared Error (RMSE):  353362.52750239917
Mean Absolute Error (MAE):  152638.39074675326
```



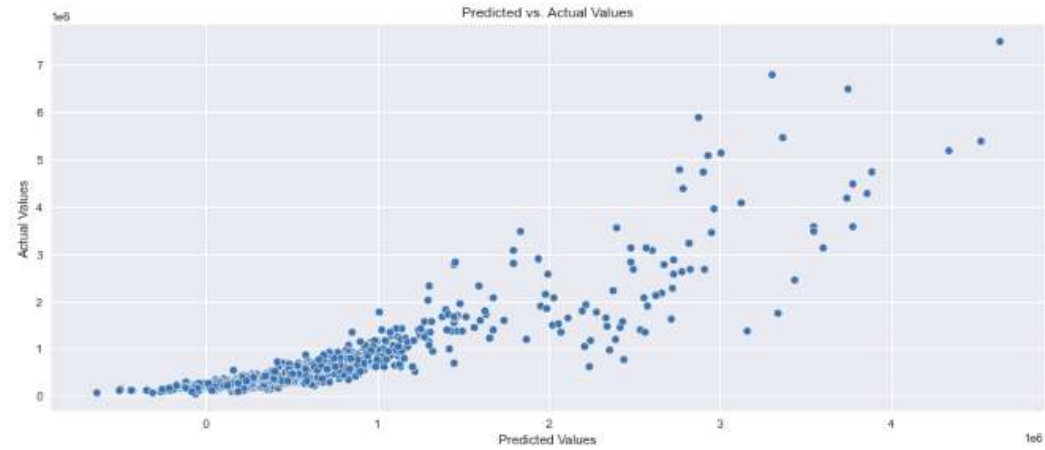

```
DecisionTreeRegressor(max_features='auto')
R Squared (R2):  85.71820756917948
Mean Squared Error (MSE):  79820820737.71753
Root Mean Squared Error (RMSE):  282525.7877393098
Mean Absolute Error (MAE):  97981.56331168831
```



```
Lasso()
R Squared (R2):  78.53383251107547
Mean Squared Error (MSE):  119974234001.72075
Root Mean Squared Error (RMSE):  346372.96950212604
Mean Absolute Error (MAE):  183979.7163302098
```




```
Ridge()
R Squared (R2):  78.53290464518219
Mean Squared Error (MSE):  119979419836.5867
Root Mean Squared Error (RMSE):  346380.45533284167
Mean Absolute Error (MAE):  183970.27969762788
```



	Model	Training Score	Test Score	Cross Validation Score	Difference
0	RandomForest	98.810911	90.883777	89.093188	1.790589
1	KNN	83.482652	77.658748	74.881360	2.777388
2	DecisionTree Regressor	99.998558	85.718208	80.630116	5.088092
3	Lasso	73.920469	78.533833	73.348394	5.185438
4	Ridge	73.920467	78.532905	73.348618	5.184286

From the above, it can be concluded the Random Forest is the best model for building the prediction model for the problem in hand.

Hyperparameter Tuning

```
param_grid = {'n_estimators':[50,100],
              'max_features':['auto','sqrt'],
              'max_depth':[4,5,None],'min_samples_split' : [2, 5, 10],
              'criterion':['squared_error','mse'],'min_samples_leaf': [1, 2, 3]}

gridsearch=GridSearchCV(estimator = rf, param_grid = param_grid,cv=5)

gridsearch.fit(x_train,y_train)
```

```
GridSearchCV(cv=5, estimator=RandomForestRegressor(),
             param_grid={'criterion': ['squared_error', 'mse'],
                         'max_depth': [4, 5, None],
                         'max_features': ['auto', 'sqrt'],
                         'min_samples_leaf': [1, 2, 3],
                         'min_samples_split': [2, 5, 10],
                         'n_estimators': [50, 100]})
```

```
gridsearch.best_score_
```

```
0.9001941155814615
```

```
gridsearch.best_params_
```

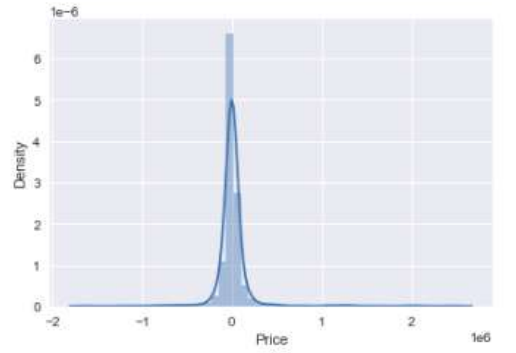
```
{'criterion': 'mse',
 'max_depth': None,
 'max_features': 'auto',
 'min_samples_leaf': 1,
 'min_samples_split': 2,
 'n_estimators': 100}
```

```
Rand_Final = RandomForestRegressor(n_estimators=100,max_features='auto',max_depth=None,criterion='mse',
                                   min_samples_split=2,min_samples_leaf=1)

Rand_Final.fit(x_train,y_train)
predictions = Rand_Final.predict(x_test)
```

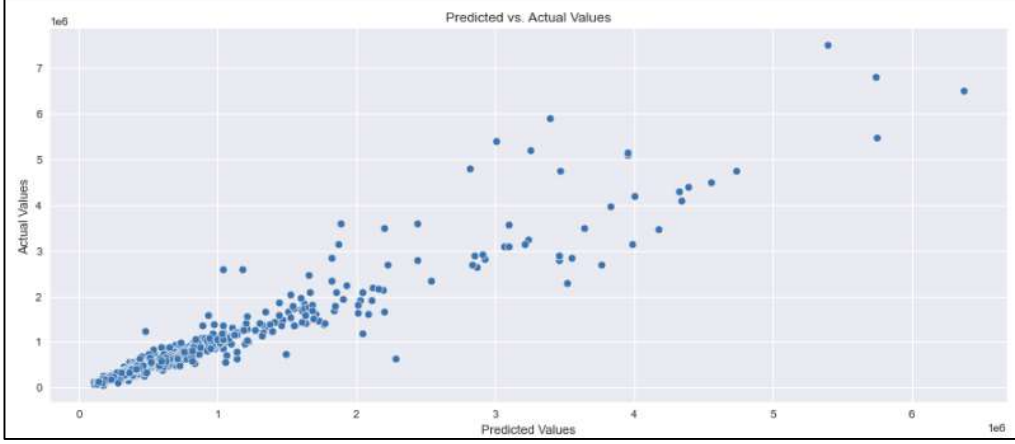
```
print('R Squared (R2): ',r2_score(y_test,predictions))
print('Mean Squared Error (MSE): ',mean_squared_error(y_test,predictions))
print('Root Mean Squared Error (RMSE): ',np.sqrt(mean_squared_error(y_test, predictions)))
print('Mean Absolute Error (MAE): ',mean_absolute_error(y_test,predictions))
```

```
R Squared (R2):  0.907346140378023
Mean Squared Error (MSE):  51784166135.78089
Root Mean Squared Error (RMSE):  227561.34587354877
Mean Absolute Error (MAE):  82233.35564631649
```

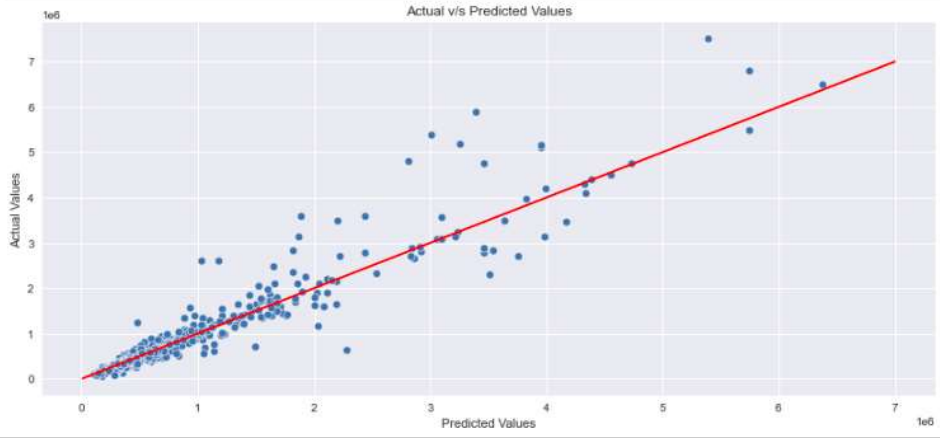
```
sns.distplot(y_test-predictions)
plt.show()
```



```
plt.figure(figsize=[15,6])
sns.scatterplot(x=predictions, y=y_test)
plt.xlabel('Predicted Values')
plt.ylabel('Actual Values')
plt.title('Predicted vs. Actual Values')
plt.show()
```




```
plt.figure(figsize=[14,6])

plot_LOO = sns.scatterplot(predictions,y_test)
plot_LOO.set(xlabel='Predicted Values', ylabel='Actual Values')

x_plot = np.linspace(10000,7000000)
y_plot = x_plot

plt.plot(x_plot, y_plot, color='r')
plt.title("Actual v/s Predicted Values")
plt.show()
```



Saving the model

```
import joblib

joblib.dump(Rand_Final,'Used_Cars_Price_Prediction.pkl')
```

```
['Used_Cars_Price_Prediction.pkl']
```

Loading the model and predicting the prices ¶

```
Model = joblib.load("Used_Cars_Price_Prediction.pkl")
```

```
Predictions = Model.predict(x_test)

Predictions
```

```
array([409766.29, 295350.  , 410034.97, ..., 579974.32, 591860.  ,
       818299.01])
```

```
list_of_tuples = list(zip(y_test, Predictions))
Used_Car_Price = pd.DataFrame(list_of_tuples, columns = ['Actual', 'Predicted'])
```

```
pd.set_option('display.float_format', '{:.2f}'.format)

Used_Car_Price['Difference'] = Used_Car_Price['Actual'] - Used_Car_Price['Predicted']
Used_Car_Price['Variance'] = (Used_Car_Price['Difference']/Used_Car_Price['Actual'])*100

Used_Car_Price
```

	Actual	Predicted	Difference	Variance
0	484999	409766.29	75232.71	15.51
1	325000	295350.00	29650.00	9.12
2	447500	410034.97	37465.03	8.37
3	395000	381054.99	13945.01	3.53
4	235500	235560.00	-60.00	-0.03
5	416500	414109.98	2390.02	0.57
6	529500	482904.81	46595.19	8.80
7	795000	812941.18	-17941.18	-2.26
8	550000	637847.02	-87847.02	-15.97

Key Metrics for Success in Solving Problem under consideration

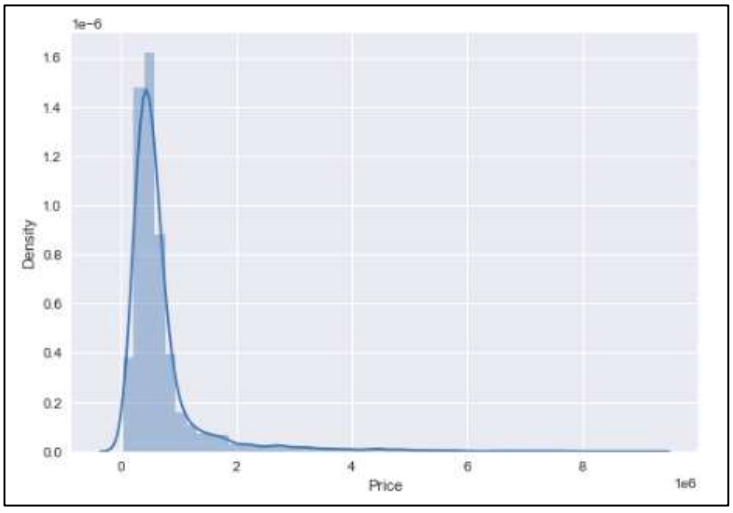
The crucial steps in any machine learning prediction model are to compute the accuracy and document the metrics on the error rates of the model. The following metrics were used:

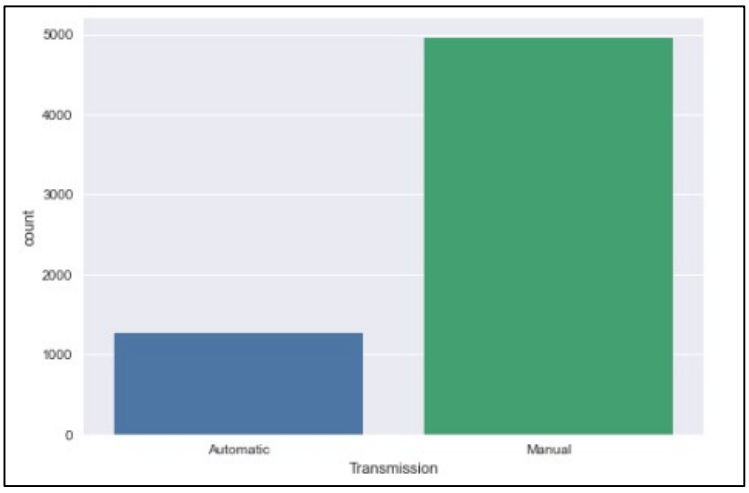
1. **Mean Absolute Error (MAE)**: It is a popular error metric for regression problems which gives magnitude of absolute difference between actual and predicted values.
2. **Mean Squared Error (MSE)**: It is a most used and very simple metric with a little bit of change in mean absolute error. Mean squared error states that finding the squared difference between actual and predicted value.
3. **Root Mean Squared Error (RMSE)**: RMSE is an extension of the mean squared error. The square root of the error is calculated, which means that the units of the RMSE are the same as the original units of the target value that is being predicted.
4. **R^2 Score**: It is the proportion of the variation in the dependent variable that is predictable from the independent variable.
5. **Cross Validation Score**: Cross-validation is primarily used in applied machine learning to estimate the skill of a machine learning model on unseen data. That is, to use a limited sample to estimate how the model is expected to perform in general when used to make predictions on data not used during the training of the model.

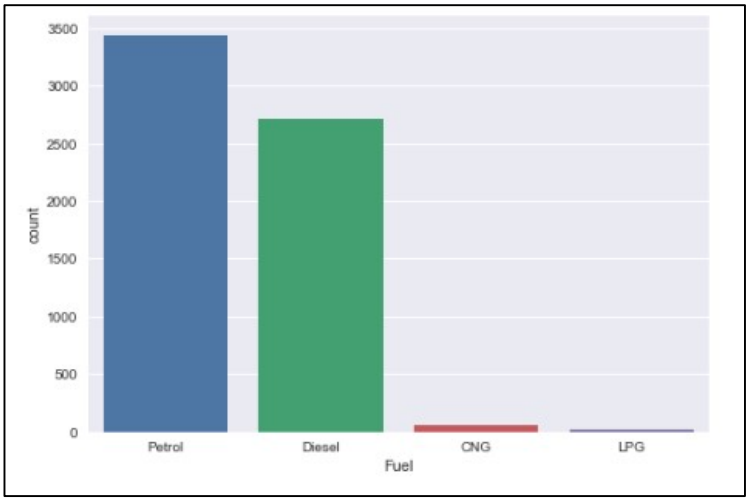
Visualizations

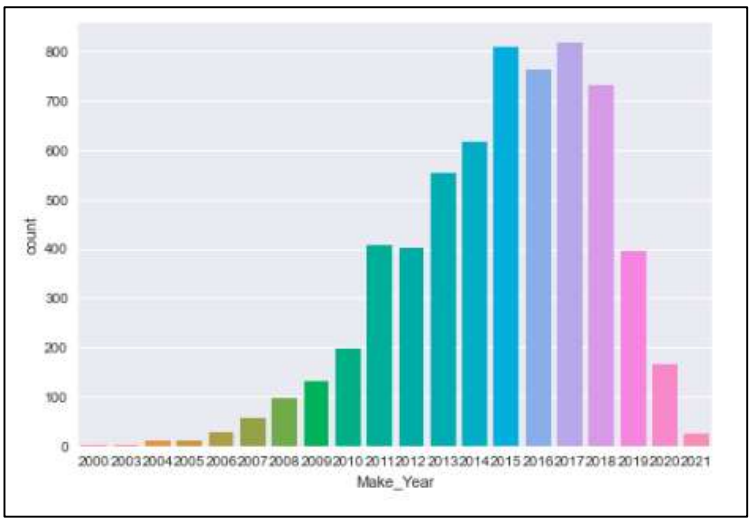
I used univariate, bivariate, and multivariate analysis to analyse the data. I used distribution plots and count plots in univariate analysis, and catplots in bivariate analysis to understand the relationship between categorical variables and the target column Car price. I also plotted a heatmap to conduct multivariate analysis and understand relationships of variables with each other.

Univariate analysis and visualizations



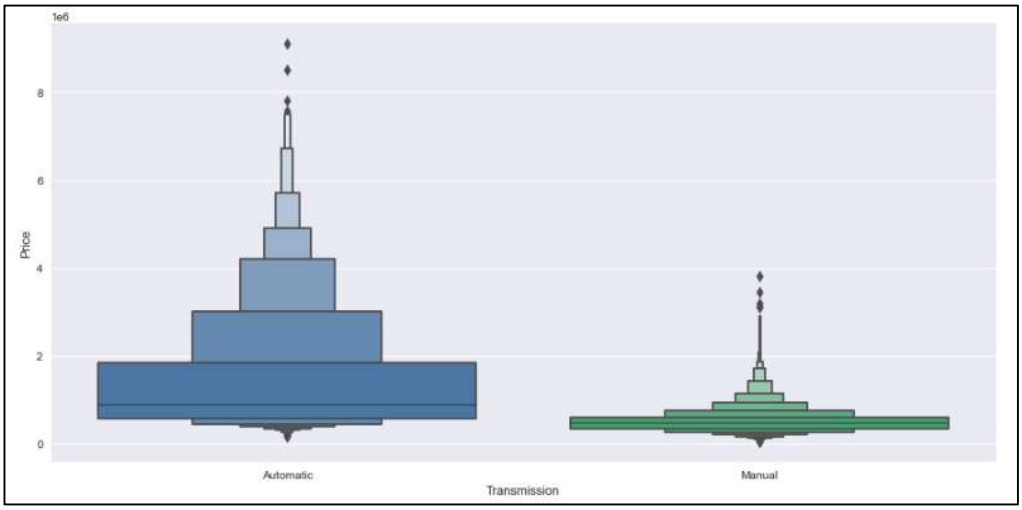


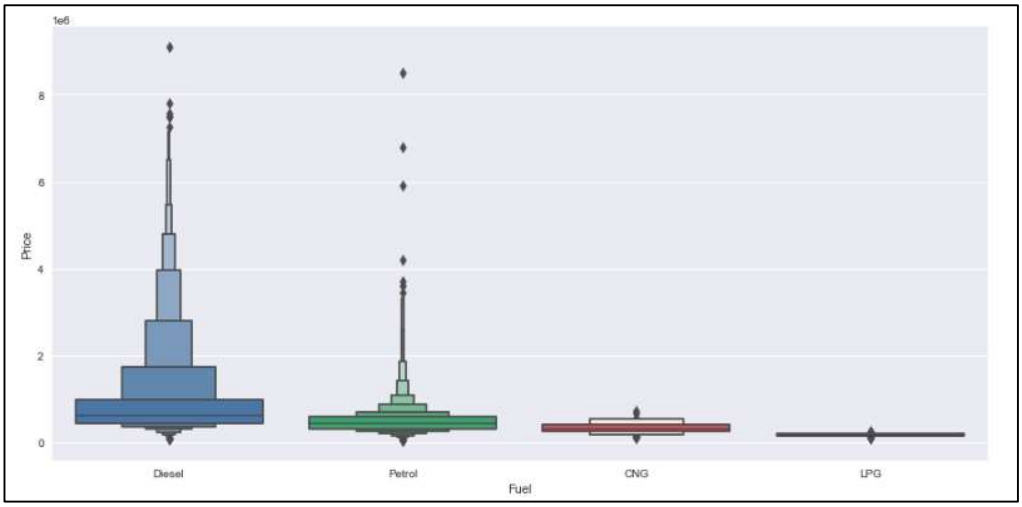


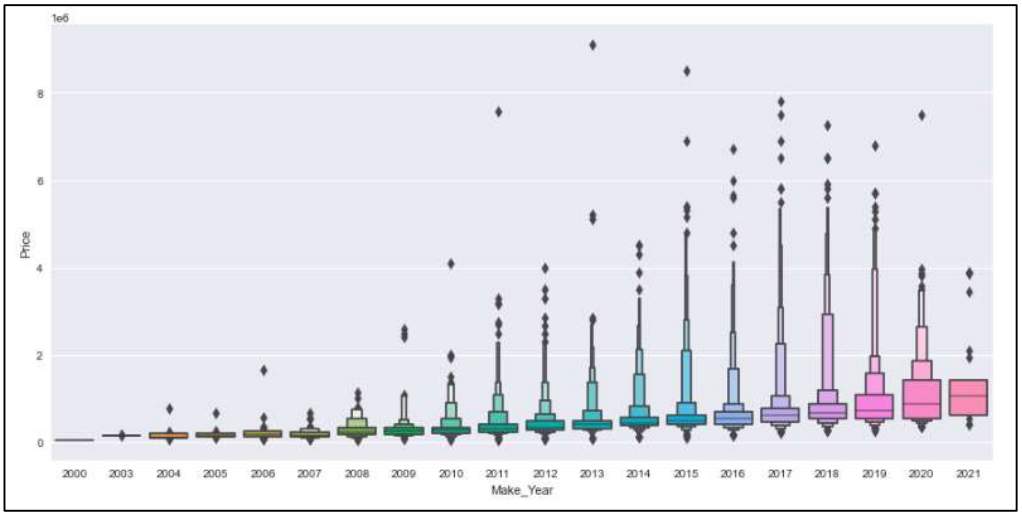


Findings: Price is not normally distributed across the dataset. Most cars available for sale on the website are of the manual transmission. Cars with petrol and diesel as fuel type were readily available for sale, with comparatively lower numbers for both, CNG and LPG.

Bivariate analysis and visualizations



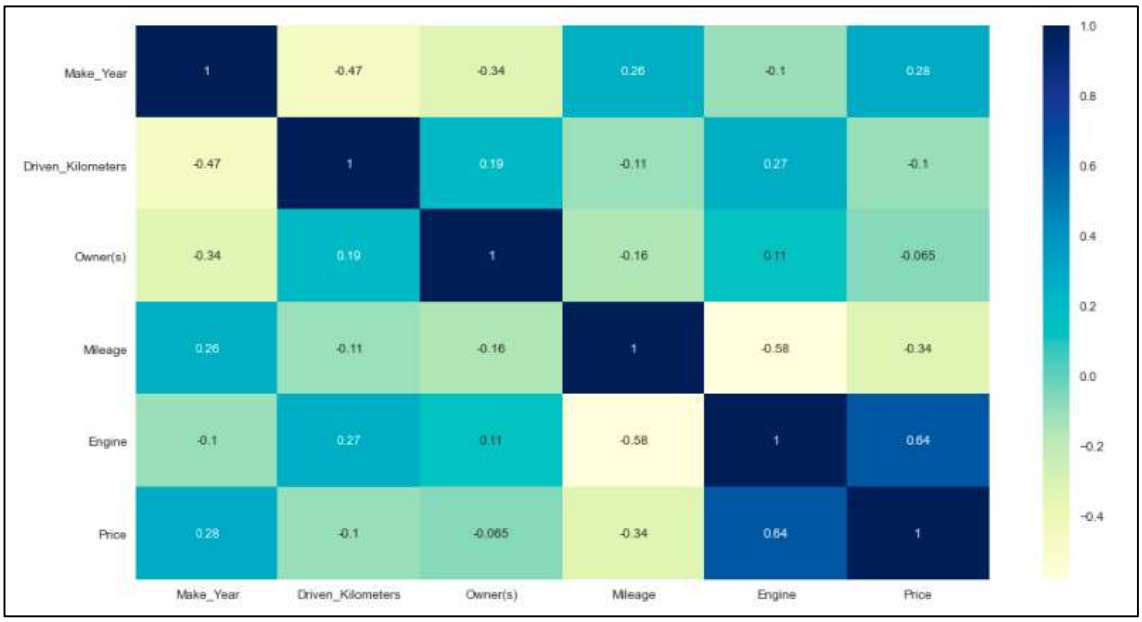




Findings: Price of automatic transmission cars is higher than that of manual transmission cars. Cars running on diesel were more expensive than any other fuel type, followed by those running on petrol, CNG and LPG respectively. From the third plot, it is evident that prices were consistently growing for

models manufactured till 2019, post which, the resale prices declined drastically, leading to a conclusion that the global pandemic left consumers with wiped out disposable incomes and savings, and cars purchased during the pandemic were possibly being sold under financial distress. To test this hypothesis, a more detailed research / study may be required.

Multivariate analysis and visualizations



A correlation matrix is essentially a table that shows the relationship between two or more variables. The measure works best with variables that have a linear relationship with one another. A heatmap can be used to visualise how well the data fits together.

Findings: Engine and Price were highly correlated, whereas, surprisingly, number of owners and price had the least correlation.

Interpretation of the Results

Visualizations: Univariate analysis helped to visualise the counts in categorical and numerical variables. Bivariate visualizations checked for the relationship between label and the features. The heatmap helped with the understanding of correlation between dependent and independent features and helped to check for the presence of multicollinearity and understand feature importance. Outliers were detected with the use of boxplots, which were removed using the IQR method. Distribution and skewness were checked using distplots.

Pre-processing: To get appropriate predictions, the dataset should be cleaned and scaled before building the ML models. I went through a few processing processes, as I outlined in the pre-processing steps, to ensure that all the relevant features were present in the dataset and that it was ready for model creation.

Model building: After cleaning and processing data, I performed train test split to build the model. I have built multiple regression models to get the accurate R2 score, and evaluation metrics like MAE, MSE and RMSE. I found that random forest regressor was the best model for prediction for the given problem and post the hyperparameter tuning, the model was saved and used for carrying out the testing of the predictions.

CONCLUSION

Key Findings and Conclusions of the Study

We created multiple regression models in this research to forecast the selling price of cars based on some of the cars' characteristics. We compared and assessed each model to see which one performed the best. We also investigated how various models order the attributes in terms of relevance. We followed the data science process in this study, starting with gathering data, cleaning, and pre-processing it, then examining the data and developing models, and finally evaluating the results.

As a recommendation, we encourage that automobile buyers use this model (or a variation of it trained with more recent data) to estimate car prices. The model can be utilised with datasets that cover many areas if they have the same properties. We also recommend that individuals evaluate the elements that were deemed the most significant in the preceding section, as this may help them better estimate the car price.

We examined the CV score after training the model to make sure it wasn't overfitting. The best model's R2 score rose after hyper parameter adjustment, and the best model's R2 score was 90.73 percent. We've also gotten some good automobile price forecasting findings.

We learned throughout the entire investigation that continuous numerical variables had a strong positive linear relationship with the label "Car Price." We discovered that automobiles with automatic gear transmission, cars utilising petrol and diesel as fuels have high sale prices by comparing car prices and categorical characteristics.

When comparing continuous numerical variables to Car Price, we discovered that automobiles with good mileage, engine displacement, and less running in kilometres had a solid linear relationship with the price, implying that cars with these features sell for a high price. We identified and removed the outliers, as well as skewness. Looking at the heat map, I identified features that were highly correlated, and scaled the data to compensate for the data bias.

Learning Outcomes of the study in respect of Data Science

I learned a lot about automotive features and car selling web platforms while working on this project, as well as how machine learning models have helped predict the price of used cars, giving me a better understanding of the various reasons and benefits of selling old cars. The project piqued my interest because the dataset contains a wide range of data types. To visualise the relationship between target and features, I used a variety of plotting techniques. This graphical representation assisted me in comprehending which features are important and how these features describe the selling price of used cars. Data cleaning was one of the most important and critical aspects of this project, in which I dealt with features with string values, feature extraction, and feature selection. Finally, Random Forest regressor turned out to be the best model.

One of the difficulties I encountered while working on this project was the time it took to scrape real-time data from the Cardekho website. The data was difficult to manage, and the cleaning process was difficult for me, but I succeeded.

Finally, we accomplished our goal by predicting the sale price of used cars and developing a car price evaluation model that could assist clients in understanding the future price of used cars.

Limitation of this work and Scope for Future Work

The study's main limitation is the small number of records that were used. Our data is not evenly distributed in some of the columns in the dataset, and some of them are unrealistic.

To anticipate car costs in the future, we plan to collect more data and apply more complex approaches such as artificial neural networks and genetic algorithms. In the future, our machine learning model might be linked to a variety of websites that provide real-time data for price forecasting. We might also include a lot of previous car price data to help improve the machine learning model's accuracy.