USED CAR PRICE PREDICTION

MACHINE LEARNING

REGRESSION PROBLEM

Submitted by:

RICHARD PRABHAKAR

ACKNOWLEDGMENT

This includes mentioning of all the references, research papers, data sources, professionals and other resources that helped you and guided you in completion of the project.

Links:-

https://stackoverflow.com/questions/72677752/create-x-train-and-y-train-for-csv-dataset-in-python

https://stackoverflow.com/questions/68794590/how-should-i-predict-target-variable-if-it-is-not-included-in-the-test-data-for

https://www.kaggle.com/datasets/nehalbirla/vehicle-dataset-from-cardekho

Medium.com- blogs on EDA

EDA by Analytics Vidya

Regularized Regression Models Kaggle notebook – by CHOWDHURY SALEH AHMED RONY, KERIMCAN ARSLAN, JAKKI SESHAPANPU

https://www.kaggle.com/code/spscientist/a-simple-tutorial-on-exploratory-data-analysis

Geeksforgeeks.com- Dataframe manipulation

Blogs on current automobile industry: -

https://www.autocarindia.com/auto-blogs/how-to-save-big-despite-no-new-car-discount-427176

https://www.autocarindia.com/auto-blogs/opinion-highlights-of-2022-426789

https://www.autocarindia.com/auto-blogs/opinion-are-indians-saving-spending-or-splurging-426594

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 have to make car price valuation model.

- ## Conceptual Background of the Domain Problem

The problem of predicting used car prices involves the use of algorithms and statistical models to predict the value of a used car based on a set of features. The features may include the make and model of the car, its age, mileage, condition, and other relevant information.

The goal of this problem is to build a model that can accurately estimate the value of a used car, so that buyers and sellers can make informed decisions about the price of a car. To do this, machine learning algorithms are trained on large datasets of used car prices, using the features of each car as input and the corresponding price as output.

This problem is challenging because used car prices can be influenced by a wide range of factors, such as market conditions, local economic conditions, and the specific features and condition of the car. Therefore, a good model must be able to capture the complex relationships between these factors and used car prices.

The use of machine learning for this problem is attractive because it allows for the automation of the prediction process, and can also provide insights into the relative importance of different features for used car pricing. This can be useful for both buyers and sellers, as well as

for car dealerships, who can use these insights to optimize their pricing strategies.

- Review of Literature

In order to build a used car price prediction model using data science and machine learning, a review of the literature is an important first step. The review of literature will help you understand the current state of the field, identify gaps in the existing knowledge, and inform the design of your model. Here are some of the key aspects of the literature review that should be considered:

- Previous studies on used car price prediction: A review of the existing literature on used car price prediction will give you an idea of what has been done before and what approaches have been used to address this problem. This will help you understand the strengths and weaknesses of previous models, and identify any opportunities for improvement.
- Feature selection and data pre-processing: A key aspect of building a used car price prediction model is selecting the right features to include in the model. The literature review should consider the importance of different features for used car pricing and the methods that have been used to pre-process the data.
- Machine learning algorithms: A review of the literature should also consider the various machine learning algorithms that have been used for used car price prediction, including linear regression, decision trees, random forests, gradient boosting, and neural networks. This will help you determine which algorithms are best suited for this problem and how they can be optimized and improved.
- Performance evaluation: Finally, the literature review should consider the methods used to evaluate the performance of used car price prediction models, including accuracy metrics, cross-validation, and model comparison. This will help you to determine the best approach for evaluating your own model and comparing it to existing models.

Overall, the literature review should provide a comprehensive overview of the field and help you to build a solid foundation for your used car price

prediction model. By considering the key aspects of the literature review, you can ensure that your model is well-informed, based on the latest research and best practices, and optimized for performance.

- ## Motivation for the Problem Undertaken

  The motivation for building a used car price prediction model with data science and machine learning is to help both buyers and sellers make informed decisions about the value of used cars. Accurately predicting the value of a used car can be challenging due to the complexity of the market and the many factors that influence used car prices. By using data science and machine learning techniques, it is possible to automate the prediction process and provide more accurate and reliable estimates of used car prices.

- For buyers, an accurate used car price prediction model can help them to make informed purchasing decisions, by providing them with a better understanding of the value of a particular car and helping them to identify cars that are overpriced or under-priced.

- For sellers, a used car price prediction model can help them to set a fair price for their car, by providing them with an estimate of its value based on its features and condition. This can help sellers to maximize their profit and avoid over- or under-pricing their car.

- In addition, the development of a used car price prediction model has the potential to provide valuable insights into the used car market and the factors that influence used car prices. This can help car manufacturers and dealers to make informed decisions about their products and services, and can also provide valuable information for governments and regulatory bodies that are interested in understanding the used car market.

- Overall, the motivation for building a used car price prediction model with data science and machine learning is to provide a useful tool for buyers, sellers, and market participants, and to contribute to a better understanding of the used car market and the factors that influence used car prices.

Analytical Problem Framing

- Mathematical/ Analytical Modeling of the Problem

  Various Statistical modelling used and the concept behind their use-case here:-

  Before model building , we used basic statistical tolls to make analytical decisions to do feature engineering

  The Tools we used are mean , median , mode, study of quantile's. we used skewness to see if the data was normally distributed, and we used correlation to see if there is similar relationship between the independent features and we used it to see which features are more related to the label. We also used box-cox which is a transformation technique used to remove outliers and then we started using the models such as :-

  1. Linear regression: This is a basic and widely used technique for modeling the relationship between a dependent variable and one or more independent variables. It involves fitting a straight line (or hyperplane in higher dimensions) to the data such that the distance between the data points and the line is minimized. Linear regression can be used for both simple linear regression (one independent variable) and multiple linear regression (more than one independent variable).
     Reason: as we are predicting a continues target variable
  2. Decision tree regression: This is a type of non-linear regression that involves building a decision tree to make predictions. It is useful for modeling relationships between variables that are not linear.
     Reason: As we are deriving the best features of a house we need to predict the price of a house , the decision tree can narrow it down
  3. Random forest regression: This is an ensemble method that involves training multiple decision tree models and combining

their predictions to make a final prediction. It is often more accurate than a single decision tree, but it is also more computationally expensive
Reason: Similar to Decision tree but multiple trees to get an even better model as with more tree we can cross verify and come up with the best

4. XGBoost (eXtreme Gradient Boosting) is a popular and powerful machine learning algorithm that can be used for both regression and classification tasks. It is an implementation of gradient boosting, which is a type of ensemble method that combines the predictions of multiple weak models to create a strong model.
    a. In the case of regression, XGBoost builds an ensemble of decision trees to make predictions. The algorithm works by training a series of decision tree models in a sequential manner, where each tree is trained to correct the mistakes made by the previous tree. The predictions of all the trees are then combined to make a final prediction.
    b. There are several mathematical, statistical, and analytical techniques that are used in XGBoost regression:
    c. Boosting: As mentioned above, XGBoost is an implementation of gradient boosting, which is a type of boosting algorithm. Boosting algorithms work by training a series of weak models sequentially, where each model is trained to correct the mistakes made by the previous model. The predictions of all the models are then combined to make a final prediction.
    d. Decision trees: XGBoost uses decision trees as the base model for its ensemble. Decision trees are a type of non-linear model that can be used for both regression and classification tasks. They work by dividing the

feature space into regions and making predictions based on which region the data belongs to.

e. Gradient descent: XGBoost uses gradient descent to optimize the loss function and find the optimal values for the model parameters. Gradient descent is an iterative optimization algorithm that works by moving in the direction of the negative gradient of the loss function, which helps to minimize the loss.
f. Regularization: XGBoost includes several regularization techniques that can help to prevent overfitting, including L1 and L2 regularization and early stopping. These techniques help to reduce the complexity of the model and improve its generalization performance.
g. Feature importance: XGBoost includes a feature importance feature that can be used to identify which features are most important for making predictions. This can be useful for feature selection and understanding the underlying relationships in the data.

Reason: One of the best models which as explained above uses many forms of statistical models and get the best or the score with the lowest error to make an effective predictor

- Data Sources and their formats

Data was scrapped from Quikr website on used cars in India alone

- Data contains 8794 entries each having 6 variables.
- Derivations of the columns in the dataset:_

```
#   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   name         8794 non-null   object
 1   company      8794 non-null   object
 2   year         8794 non-null   object
 3   Price        8794 non-null   object
 4   kms_driven   8740 non-null   object
 5   fuel_type    8741 non-null   object
```

- Data Pre-processing Done

The Steps followed in order to clean the data

1. names are pretty inconsistent
2. names have company names attached to it
3. some names are spam like 'Maruti Ertiga showroom condition with' and 'Well mentained Tata Sumo'
4. company: many of the names are not of any company like 'Used', 'URJENT', and so on.
5. year has many non-year values
6. year is in object. Change to integer
7. Price has Ask for Price
8. Price has commas in its prices and is in object
9. kms_driven has object values with kms at last.
10. It has nan values and two rows have 'Petrol' in them
11. fuel_type has nan values
12. name and company had spammed data...but with the previous cleaning, those rows got removed
13. Resetting the index of the final cleaned data
14. Removing the duplicate data
15. Only considering price of cars till 60-70 lakhs as it is illogical to sell higher price car in quikr and can be sold in showroom itself.
16. Converting the categorical columns to numerical using one hot encoder

Finally we end up with 8718 rows loss of less than 2-3% of data

- Data Inputs- Logic- Output Relationships

  The relationship between various inputs such as the year, price, kilometers driven, name of the car, manufacturing company, fuel type, and the price of a used car can be complex and dependent on many

factors. Here's a general overview of how these inputs may be related to the price of a used car:

- Year: The year in which the car was manufactured is an important factor that can affect its price. As a car ages, it is likely to depreciate in value, although certain classic and luxury vehicles may increase in value over time.
- Price: The price of a used car is the most important factor in determining its value, and it is influenced by a number of other inputs. The price of a used car can be influenced by factors such as its age, condition, mileage, and features, as well as market demand and supply.
- Kilometers driven: The number of kilometers driven is a key factor that can affect the price of a used car. A car that has been driven fewer kilometers is generally considered to be in better condition than a car that has been driven more, and is therefore likely to command a higher price.
- Name of the car: The name of the car can also have an impact on its price, as certain car brands are more popular or prestigious than others. A luxury car from a well-known brand is likely to command a higher price than a car from a lesser-known brand, even if the two cars are similar in other ways.
- Manufacturing company: The manufacturing company can also have an impact on the price of a used car. Certain car manufacturers have a reputation for producing reliable and high-quality vehicles, which can command a higher price on the used car market.
- Fuel type: The fuel type can also have an impact on the price of a used car. Cars that run on alternative fuels, such as electric or hybrid vehicles, are becoming increasingly popular and may command a higher price on the used car market.
- The logic and output relationships between these inputs and the price of a used car can be determined through

the use of machine learning algorithms, such as regression analysis or decision trees. These algorithms can analyze large amounts of data to identify patterns and relationships between the inputs and the price of a used car, and then use this information to make predictions about the price of new cars based on their inputs.

Overall, the relationships between the inputs and the price of a used car are complex and can be influenced by many factors. By using data science and machine learning techniques, it is possible to build a model that can accurately predict the price of used cars based on these inputs.

Studies from the data and the EDA done :

- We see that the year is left skewed and the price and kms driven are right skewed meaning the data is fairly new and old year is lesser and there are ouliers in price and kms driven.
- Maruthi has the highest number of cars in the dta set almost 80-90%
- Most of the cars are petrol and diesel and there are very very few lpg and electric cars
- In relationship graph between company and price , we see that benz and bmw cars have the highest range and volvo as well comes in very close
- In the same relation graph with year we see that 2016 to 2020 has the highest no of cars
- In the kms drive we see that the lower the kms driven the higher the price and the more it has the price decreases which means it has a negative relationship with price
- Fuel type as petrol and diesel have the highest more than 95% of data we see the highest range in them ‘

- Hardware and Software Requirements and Tools Used

Listing

```
import pandas as pd ## pandas is used to manupulate the dataframe

import numpy as np ## numpy is used to do scientific calculations

import matplotlib.pyplot as plt ## matplotlib used for visualization or graphs

import seaborn as sns ## seaborn used for visualization or graphs

import missingno as msno ## used to visualize missing values

import warnings ## used to remove warnings

warnings.filterwarnings('ignore')

%matplotlib inline


from sklearn.model_selection import train_test_split,cross_validate, GridSearchCV – ## used to split training data and test , in this case we didn’t use as we have already separate files for training and testing

from sklearn.preprocessing import StandardScaler,OrdinalEncoder# to convert or encode the categorical or string values into numbers

from sklearn.metrics import mean_squared_error #it is a metric used to check the error we get with each model , lower the better

from sklearn.linear_model import LinearRegression,Lasso,Ridge,BayesianRidge ## linear regression model used to predict and train data
```

```
from sklearn.ensemble import GradientBoostingRegressor, RandomForestRegressor # Ensemle techniques used to work on model to see which is better at prediction and lower error

from xgboost import XGBRegressor # another machine learning model but boosting techniques

from lightgbm import LGBMRegressor # another Boosting technique


import math #to do mathematics based functions on the data be it for visualization or for any cleaning as well

from IPython.display import Image # to save and show image

import warnings # to remove the warnings to show clean outputs

warnings.filterwarnings("ignore")

sns.set(rc={"figure.figsize": (20, 15)})

sns.set_style("whitegrid")
```

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

Also explained in statistical modelling and analytical in previous Analytical Problem Framing Chapter, we use the following methods:-

1. Linear regression: This is a basic and widely used technique for modeling the relationship between a dependent variable and one or more independent variables. It involves fitting a straight line (or hyperplane in higher dimensions) to the data such that the distance between the data points and the line is minimized. Linear regression can be used for both simple linear regression (one independent variable) and multiple linear regression (more than one independent variable).
   Reason: as we are predicting a continues target variable
2. Decision tree regression: This is a type of non-linear regression that involves building a decision tree to make predictions. It is useful for modeling relationships between variables that are not linear.
   Reason: As we are deriving the best features of a house we need to predict the price of a house , the decision tree can narrow it down
3. Random forest regression: This is an ensemble method that involves training multiple decision tree models and combining their predictions to make a final prediction. It is often more accurate than a single decision tree, but it is also more computationally expensive
   Reason: Similar to Decision tree but multiple trees to get an even better model as with more tree we can cross verify and come up with the best

4. XGBoost (eXtreme Gradient Boosting) is a popular and powerful machine learning algorithm that can be used for both regression and classification tasks. It is an implementation of gradient boosting, which is a type of ensemble method that combines the predictions of multiple weak models to create a strong model.
    a. In the case of regression, XGBoost builds an ensemble of decision trees to make predictions. The algorithm works by training a series of decision tree models in a sequential manner, where each tree is trained to correct the mistakes made by the previous tree. The predictions of all the trees are then combined to make a final prediction.
    b. There are several mathematical, statistical, and analytical techniques that are used in XGBoost regression:
    c. Boosting: As mentioned above, XGBoost is an implementation of gradient boosting, which is a type of boosting algorithm. Boosting algorithms work by training a series of weak models sequentially, where each model is trained to correct the mistakes made by the previous model. The predictions of all the models are then combined to make a final prediction.
    d. Decision trees: XGBoost uses decision trees as the base model for its ensemble. Decision trees are a type of non-linear model that can be used for both regression and classification tasks. They work by dividing the feature space into regions and making predictions based on which region the data belongs to.
    e. Gradient descent: XGBoost uses gradient descent to optimize the loss function and find the optimal values for the model parameters. Gradient descent is an iterative optimization algorithm that works by moving in the direction of the negative gradient of the loss function, which helps to minimize the loss.

f. Regularization: XGBoost includes several regularization techniques that can help to prevent overfitting, including L1 and L2 regularization and early stopping. These techniques help to reduce the complexity of the model and improve its generalization performance.
g. Feature importance: XGBoost includes a feature importance feature that can be used to identify which features are most important for making predictions. This can be useful for feature selection and understanding the underlying relationships in the data.

Reason: One of the best models which as explained above uses many forms of statistical models and get the best or the score with the lowest error to make an effective predictor

- Testing of Identified Approaches (Algorithms)

Linear Regression alone and with Tuning techniques,Ridge Regression,Random forest Regressor with and withput hyper parameter tuning techniques,Decision tree Regressor with and without tuning,Xgboost Regressor with and without tuning

- Run and Evaluate selected models
  Snapshot of results with all models :-

Linear Regression

In [133]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
from sklearn.metrics import r2_score, mean_squared_error, mean_absolute_error
from sklearn.model_selection import train_test_split
```

In [164]:
```
scores_test=[]
for i in range(0,100):
    X_train,X_test,y_train,y_test = train_test_split(X,y,test_size = 0.25,random_state = i)
    lr.fit(X_train,y_train)
    pred_train = lr.predict(X_train)
    pred_test = lr.predict(X_test)
    print(f"At random state {i},the training accuracy is :-{r2_score(y_train,pred_train)}")
    print(f"At random state {i},the testing accuracy is :-{r2_score(y_test,pred_test)}")
    print('\n')
    scores_test.append(r2_score(y_test,pred_test))
```

We see the following states giving desirable scores

- At random state 86,the training accuracy is :-0.8751759620343398
- At random state 86,the testing accuracy is :-0.7872334530832948
- At random state 84,the training accuracy is :-0.8365708488486432
- At random state 84,the testing accuracy is :-0.8009693192582444
- At random state 86,the training accuracy is :-0.8319015551760935
- At random state 86,the testing accuracy is :-0.8108931549841014

Best score as per the analysis for Linear regression is at state 86

At random state 86,the training accuracy is :-0.8319015551760935

At random state 86,the testing accuracy is :-0.8108931549841014

In [169]:
```
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size = 0.25,random_state = 86)
```

In [170]:
```
lr.fit(X_train,y_train)
```

Out[170]:
```
LinearRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [171]:
```
pred_test=lr.predict(X_test)
```

In [172]:
```
print(r2_score(y_test,pred_test))
```

```
0.8108931549841014
```

Cross-Validation of the model

In [173]:
```
Train_accuracy=r2_score(y_train,pred_train)
Test_accuracy=r2_score(y_test,pred_test)

from sklearn.model_selection import cross_val_score
for j in range(2,10):
    cv_score=cross_val_score(lr,X,y,cv=j)
    cv_mean=cv_score.mean()
    print(f"At cross fold{j} the cv score is {cv_mean} and accuracy score for training is {Train_accuracy}and the accuracy for t
    print('\n')
```

```
At cross fold2 the cv score is 0.7282091418073493 and accuracy score for training is -0.838294735269473and the accuracy for tes
ting is 0.8108931549841014


At cross fold3 the cv score is -4507804928.444499 and accuracy score for training is -0.838294735269473and the accuracy for tes
ting is 0.8108931549841014


At cross fold4 the cv score is -8035892193.307001 and accuracy score for training is -0.838294735269473and the accuracy for tes
ting is 0.8108931549841014


At cross fold5 the cv score is -13489665139.187572 and accuracy score for training is -0.838294735269473and the accuracy for te
sting is 0.8108931549841014


At cross fold6 the cv score is -1290808635.8864625 and accuracy score for training is -0.838294735269473and the accuracy for te
sting is 0.8108931549841014


At cross fold7 the cv score is -3777478034.2262607 and accuracy score for training is -0.838294735269473and the accuracy for te
sting is 0.8108931549841014
```

Regularization of the Linear Model

In [176]:
```
from sklearn.model_selection import GridSearchCV #to select the best parameters for hyperparameter tuning
from sklearn.model_selection import cross_val_score #to check the difference from the earlier score without hyper parameter tunir
```

In [177]:
```
from sklearn.linear_model import Lasso

parameters ={'alpha' : [.0001, .001, .01, .1, 1, 10],
             'random_state' : list(range(0,15))}
ls = Lasso()
clf = GridSearchCV(ls,parameters)
clf.fit(X_train, y_train)

print(clf.best_params_)
```

```
{'alpha': 10, 'random_state': 0}
```

Final model training for Linear Regression

In [179]:
```
ls = Lasso(alpha= 10, random_state= 0)
ls.fit(X_train,y_train)
ls_score_training = ls.score(X_train,y_train)
pred_ls = ls.predict(X_test)
ls_score_training*100
```

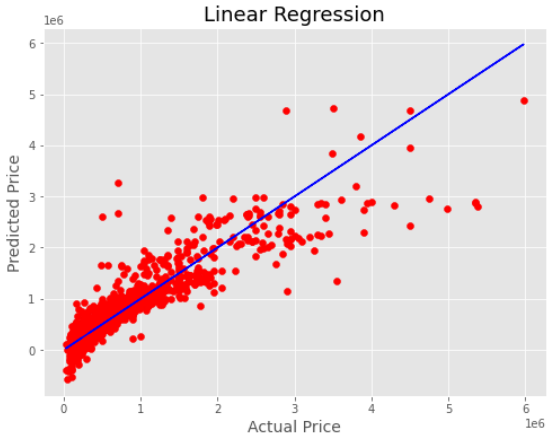
Out[179]:
```
83.05976298557103
```

we have increased the test score from 81 to 83

At cross fold2 the cv score is 0.7282091418073493 and accuracy score for training is -0.838294735269473and the accuracy for testing is 0.8108931549841014

Plotting the linear Regression graph with actual and predicted values comparison

n [175]:
```
import matplotlib.pyplot as plt
plt.figure(figsize=(8,6))
plt.scatter(x=y_test, y=pred_test,color='r')
plt.plot(y_test,y_test,color='b')
plt.xlabel('Actual Price',fontsize=14)
plt.ylabel('Predicted Price',fontsize=14)
plt.title('Linear Regression',fontsize=18)
plt.show()
```



Checking MSE,RMSE score

In [180]:
```
from sklearn import metrics

print('MAE:', metrics.mean_absolute_error(y_test, pred_test))
print('MSE:', metrics.mean_squared_error(y_test, pred_test))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, pred_test)))
```

```
MAE: 148333.37078024718
MSE: 75539730869.94919
RMSE: 274844.92149201006
```

we see that the MAE,MES,RMSE metrics are really bad with the linearmodel so we will test other models ,

Decision Tree Regressor

In [184]:
```
from sklearn.tree import DecisionTreeRegressor

dt=DecisionTreeRegressor()
dt.fit(X_train,y_train)
dt.score(X_train,y_train)
pred_test =dt.predict(X_test)
dfs = r2_score(y_test,pred_test)
print('R2 Score :',dfs*100)

dfscore = cross_val_score(dt,X,y,cv=2)
dfc =dfscore.mean()
print('Cross Val Score :',dfc*100)
```

```
R2 Score : 78.40260096204126
Cross Val Score : 77.41114311167591
```

In [66]:
```
rf_random.fit(X_train,y_train)
```

```
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:   12.9s remaining:    0.0s

[CV]  n_estimators=900, min_samples_split=5, min_samples_leaf=5, max_features=sqrt, max_depth=10, total=  11.5s
[CV] n_estimators=900, min_samples_split=5, min_samples_leaf=5, max_features=sqrt, max_depth=10
[CV]  n_estimators=900, min_samples_split=5, min_samples_leaf=5, max_features=sqrt, max_depth=10, total=  11.8s
[CV] n_estimators=900, min_samples_split=5, min_samples_leaf=5, max_features=sqrt, max_depth=10
[CV]  n_estimators=900, min_samples_split=5, min_samples_leaf=5, max_features=sqrt, max_depth=10, total=  12.4s
[CV] n_estimators=900, min_samples_split=5, min_samples_leaf=5, max_features=sqrt, max_depth=10
[CV]  n_estimators=900, min_samples_split=5, min_samples_leaf=5, max_features=sqrt, max_depth=10, total=  10.9s
[CV] n_estimators=1100, min_samples_split=10, min_samples_leaf=2, max_features=sqrt, max_depth=15
[CV]  n_estimators=1100, min_samples_split=10, min_samples_leaf=2, max_features=sqrt, max_depth=15, total=  18.6s
[CV] n_estimators=1100, min_samples_split=10, min_samples_leaf=2, max_features=sqrt, max_depth=15
[CV]  n_estimators=1100, min_samples_split=10, min_samples_leaf=2, max_features=sqrt, max_depth=15, total=  17.9s
[CV] n_estimators=1100, min_samples_split=10, min_samples_leaf=2, max_features=sqrt, max_depth=15
[CV]  n_estimators=1100, min_samples_split=10, min_samples_leaf=2, max_features=sqrt, max_depth=15, total=  18.5s
[CV] n_estimators=1100, min_samples_split=10, min_samples_leaf=2, max_features=sqrt, max_depth=15
[CV]  n_estimators=1100, min_samples_split=10, min_samples_leaf=2, max_features=sqrt, max_depth=15, total=  16.6s
[CV] n_estimators=1100, min_samples_split=10, min_samples_leaf=2, max_features=sqrt, max_depth=15
[CV]  n_estimators=1100, min_samples_split=10, min_samples_leaf=2, max_features=sqrt, max_depth=15, total=  16.6s
[CV] n_estimators=300, min_samples_split=100, min_samples_leaf=5, max_features=auto, max_depth=15
```

In [67]:
```
rf_random.best_params_
```

Out[67]:
```
{'n_estimators': 700,
 'min_samples_split': 15,
 'min_samples_leaf': 1,
 'max_features': 'auto',
 'max_depth': 20}
```

```
Cross Val Score : 77.41114311167591
```

In [182]:
```
from sklearn import metrics

print('MAE:', metrics.mean_absolute_error(y_test, pred_test))
print('MSE:', metrics.mean_squared_error(y_test, pred_test))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, pred_test)))
```

```
MAE: 129586.6152640652
MSE: 81157278977.29672
RMSE: 284881.1664138167
```

K- Nearest Neighbors

In [187]:
```
from sklearn.neighbors import KNeighborsRegressor

knn=KNeighborsRegressor()
knn.fit(X_train,y_train)
knn.score(X_train,y_train)
pred_test =knn.predict(X_test)
knns = r2_score(y_test,pred_test)
print('R2 Score :',knns*100)

knnscore = cross_val_score(knn,X,y,cv=9)
knnc =knnscore.mean()
print('Cross Val Score :',knnc*100)
```

```
R2 Score : 12.576183453079615
Cross Val Score : 8.285522657608881
```

In [186]:
```
from sklearn import metrics

print('MAE:', metrics.mean_absolute_error(y_test, pred_test))
print('MSE:', metrics.mean_squared_error(y_test, pred_test))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, pred_test)))
```

```
MAE: 334889.42632816173
MSE: 349219065709.809
RMSE: 590947.5998003621
```

Bagging Classifier

In [188]:
```
from sklearn.ensemble import BaggingRegressor
from sklearn.neighbors import KNeighborsRegressor


knn = BaggingRegressor(KNeighborsRegressor(n_neighbors=5),
                           n_estimators=9, max_samples= 0.7,
                           bootstrap=True, random_state=3, oob_score = True)
knn.fit(X_train,y_train)
knn.score(X_train,y_train)
pred_test =knn.predict(X_test)
knns = r2_score(y_test,pred_test)
print('R2 Score :',knns*100)

knnscore = cross_val_score(knn,X,y,cv=2)
knnc =knnscore.mean()
print('Cross Val Score :',knnc*100)
```

```
R2 Score : 15.212416353264436
Cross Val Score : 7.541548235357343
```

Random Forest Regressor

In [189]:
```
from sklearn.ensemble import RandomForestRegressor

rf=RandomForestRegressor()
rf.fit(X_train,y_train)
rf.score(X_train,y_train)
pred_decision =rf.predict(X_test)

rfs = r2_score(y_test,pred_decision)
print('R2 Score :',rfs*100)

rfscore = cross_val_score(rf,X,y,cv=2)
rfc =rfscore.mean()
print('Cross Val Score :',rfc*100)
```

```
R2 Score : 85.84608793733629
Cross Val Score : 81.54113096431075
```

In [190]:
```
from sklearn import metrics

print('MAE:', metrics.mean_absolute_error(y_test, pred_test))
print('MSE:', metrics.mean_squared_error(y_test, pred_test))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, pred_test)))
```

```
MAE: 329994.1063678629
MSE: 338688493758.6073
RMSE: 581969.4955567751
```

ADA Boost Regressor

In [194]:
```
from sklearn.ensemble import AdaBoostRegressor
ada = AdaBoostRegressor()

ada.fit(X_train,y_train)
ada.score(X_train,y_train)
pred_decision =ada.predict(X_test)

adas = r2_score(y_test,pred_decision)
print('R2 Score :',adas*100)

adascore = cross_val_score(ada,X,y,cv=2)
adac =adascore.mean()
print('Cross Val Score :',adac*100)
```

```
R2 Score : 25.975884950280413
Cross Val Score : 13.4872928546307
```

In [192]:
```
#Checking MAE MSE and RMSE scores
from sklearn import metrics

print('MAE:', metrics.mean_absolute_error(y_test, pred_decision))
print('MSE:', metrics.mean_squared_error(y_test, pred_decision))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, pred_decision)))
```

```
MAE: 369502.91990453145
MSE: 235793613234.91763
RMSE: 485585.84538155317
```

Xgboost Regressor

In [199]:
```
import xgboost as xgb
xgb = xgb.XGBRegressor()

xgb.fit(X_train,y_train)
xgb.score(X_train,y_train)
pred_decision =xgb.predict(X_test)

xgbs = r2_score(y_test,pred_decision)
print('R2 Score :',xgbs*100)

xgbscore = cross_val_score(xgb,X,y,cv=6)
xgbc =xgbscore.mean()
print('Cross Val Score :',xgbc*100)
```

```
R2 Score : 90.31281272911396
Cross Val Score : 86.1626219951046
```

In [196]:
```
#Checking MAE MSE and RMSE scores
from sklearn import metrics

print('MAE:', metrics.mean_absolute_error(y_test, pred_decision))
print('MSE:', metrics.mean_squared_error(y_test, pred_decision))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, pred_decision)))
```

```
MAE: 105414.22778699092
MSE: 38695982647.693596
RMSE: 196712.94478933915
```

we choose XG boost as the best model as it r2 score is the highest and had the fastest computational time as well

Hyper parameter Tuning with XG Boost Regressor

In [197]:
```
from sklearn.model_selection import RandomizedSearchCV
import xgboost as xgb
xgb = xgb.XGBRegressor()

#Creating parameters to pass in RandomizedSearchCV

parameters = {'gamma': [0.1,0.5,1,10,16,32,64],
              'learning_rate': [0.01, 0.03, 0.06, 0.1, 0.3, 0.2],
              'max_depth': [5,6,7,8,9,10],
              'n_estimators': [80,100,120,130,150,200],
              'reg_alpha': [0.1,0.5,1,10,16,32,64],
              'reg_lambda': [0.1,0.5,1,10,16,32,64]
             }

RCV = RandomizedSearchCV(xgb,parameters,verbose=2,cv=2, n_jobs = -1)
RCV.fit(X_train,y_train) #fitting data into the model
RCV.best_params_ #printing the best parameters found by the RandomizedSearchCV
```

```
Fitting 2 folds for each of 10 candidates, totalling 20 fits
```

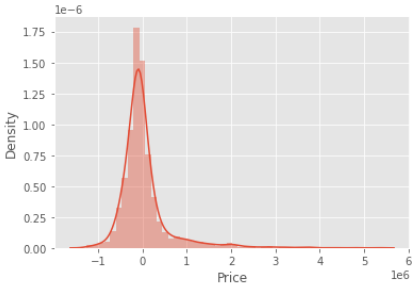
Out[197]:
```
{'reg_lambda': 1,
 'reg_alpha': 32,
 'n_estimators': 150,
 'max_depth': 8,
 'learning_rate': 0.2,
 'gamma': 10}
```

In [198]:
```
import xgboost as xgb
xgb = xgb.XGBRegressor(reg_lambda= 1,reg_alpha= 32,n_estimators= 150,max_depth= 8,learning_rate= 0.2,gamma= 10)

xgb.fit(X_train,y_train)
xgb.score(X_train,y_train)
pred_decision =xgb.predict(X_test)

xgbs = r2_score(y_test,pred_decision)
print('R2 Score :',xgbs*100)

xgbscore = cross_val_score(xgb,X,y,cv=6)
xgbc =xgbscore.mean()
print('Cross Val Score :',xgbc*100)
```

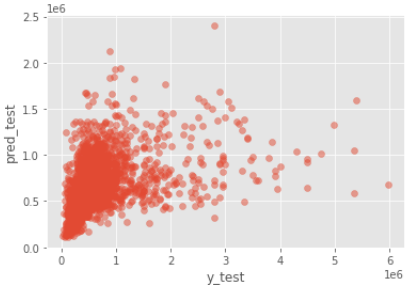
```
R2 Score : 89.8543840531255
Cross Val Score : 86.74015726630365
```

Since we are getting a lower score we will use the older models score itself

In [200]:
```
#Plotting distplot to show equillibrium

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



In [201]:
```
plt.scatter(y_test, pred_test, alpha = 0.5)
plt.xlabel("y_test")
plt.ylabel("pred_test")
plt.show()
```



In [202]:
```
import pickle
filename = 'used_carprice.pkl'
pickle.dump(xgb,open(filename,'wb'))
```

Conclusion

In [203]:
```
loaded_model = pickle.load(open('used_carprice.pkl','rb'))
result = loaded_model.score(X_test,y_test)
print(result*100)
```

```
90.31281272911396
```

In [204]:
```
conclusion = pd.DataFrame([loaded_model.predict(X_test)[:],pred_decision[:]],index=['Predicted','Orignal'])
```

In [205]:
```
conclusion
```

Out[205]:

	0	1	2	3	4	5	6	7	8	9	...	2117
Predicted	304980.03125	366227.25	210084.875	1393749.375	770231.625	661315.4375	1648171.125	927616.625	293935.46875	331395.9375	...	389342.15625
Orignal	304980.03125	366227.25	210084.875	1393749.375	770231.625	661315.4375	1648171.125	927616.625	293935.46875	331395.9375	...	389342.15625

- Key Metrics for success in solving problem under consideration

R2 Score

The R2 score, also known as the coefficient of determination, is a measure of the goodness of fit of a machine learning model. It is used to evaluate the performance of a regression model, and it can range from 0 to 1, with a higher value indicating a better fit.

The R2 score is calculated by taking the sum of the squares of the differences between the predicted values and the actual values, and dividing it by the sum of the squares of the differences between the actual values and the mean of the actual values. This results in a value between 0 and 1, with 0 indicating that the model is not a good fit and 1 indicating a perfect fit.

For example, if a model has an R2 score of 0.8, this means that the model explains 80% of the variance in the data. On the other hand, if a model has an R2 score of 0.2, this means that the model only

explains 20% of the variance in the data, which may indicate that the model is not performing well.

In general, the R2 score is a useful metric for evaluating the performance of a machine learning model, especially when the goal is to predict a continuous target variable. However, it is important to keep in mind that the R2 score can be affected by the number of variables in the model, and it may not always be the most appropriate metric to use, depending on the specific characteristics of the data and the goals of the analysis.

Result : we see that we are able to achieve 90% accuracy with Xgboost Regressor model

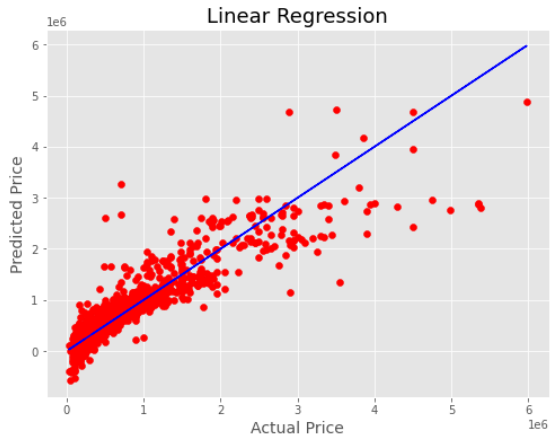
- Visualizations

Linear regression plots

```
At cross fold2 the cv score is 0.7282091418073493 and accuracy score for training is -0.838294735269473and the accuracy for testing is
0.8108931549841014
```

Plotting the linear Regression graph with actual and predicted values comparison

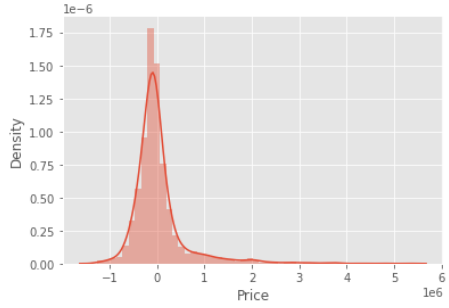
```
In [175]: import matplotlib.pyplot as plt
          plt.figure(figsize=(8,6))
          plt.scatter(x=y_test, y=pred_test,color='r')
          plt.plot(y_test,y_test,color='b')
          plt.xlabel('Actual Price',fontsize=14)
          plt.ylabel('Predicted Price',fontsize=14)
          plt.title('Linear Regression',fontsize=18)
          plt.show()
```



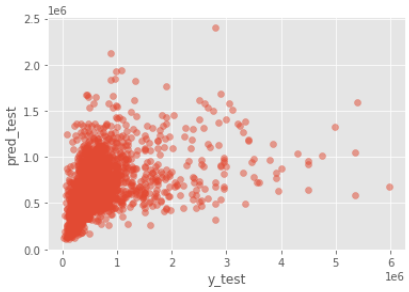
```
R2 Score : 89.8543840531255
Cross Val Score : 86.74015726630365
```

Since we are getting a lower score we will use the older models score itself

In [200]:
```
#Plotting distplot to show equillibrium

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



In [201]:
```
plt.scatter(y_test, pred_test, alpha = 0.5)
plt.xlabel("y_test")
plt.ylabel("pred_test")
plt.show()
```



In [202]:
```
import pickle
filename = 'used_carprice.pkl'
pickle.dump(xgb,open(filename,'wb'))
```

Conclusion

In [203]:
```
loaded_model = pickle.load(open('used_carprice.pkl','rb'))
result = loaded_model.score(X_test,y_test)
print(result*100)
```

```
90.31281272911396
```

In [204]:
```
conclusion = pd.DataFrame([loaded_model.predict(X_test)[:],pred_decision[:]],index=['Predicted','Orignal'])
```

In [205]:
```
conclusion
```

Out[205]:

	0	1	2	3	4	5	6	7	8	9	...	2117
Predicted	304980.03125	366227.25	210084.875	1393749.375	770231.625	661315.4375	1648171.125	927616.625	293935.46875	331395.9375	...	389342.15625
Orignal	304980.03125	366227.25	210084.875	1393749.375	770231.625	661315.4375	1648171.125	927616.625	293935.46875	331395.9375	...	389342.15625

WE SEE MUCH BETTER DISTRIBUTION IN XGBOOST REGRESSION MODEL

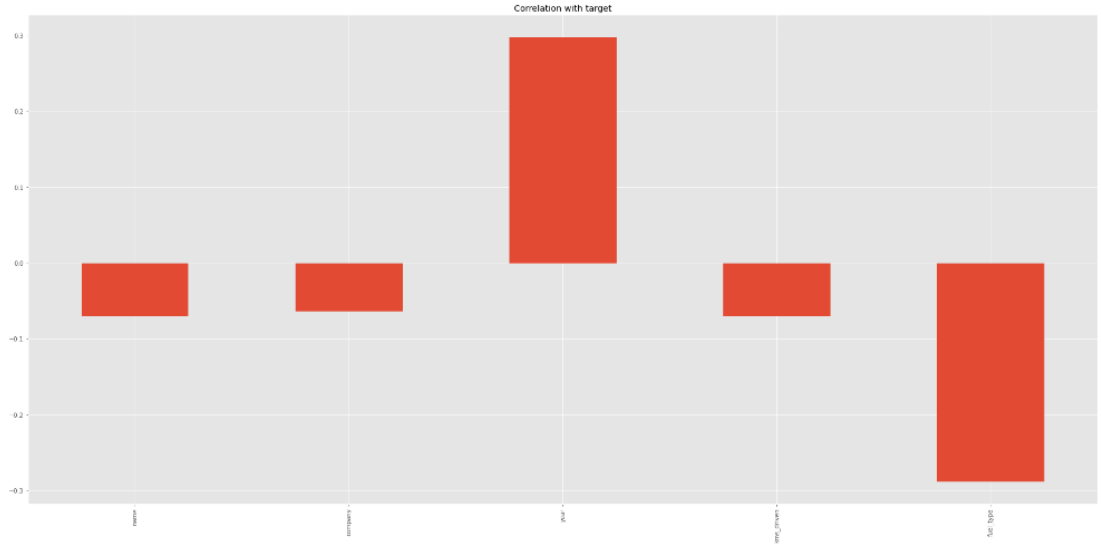
- Interpretation of the Results

From the data set we see that as we have specifically replaced the null values in many of the columns with the right replacements and we have treated the dataset for outliers therefore , we were able to get a good accuracy score , we can actually used other metrics like mean squared error as well and get 0.11 to .14% which is very good , but the accuracy score being 90% + shows that the model is very good and can be used for predictions for used car prices , we saw that few features were influencing the price way more than others

Correlation of the columns with the Price as in Label

```
In [53]: car.corr()['Price'].sort_values()

Out[53]: fuel_type    -0.288637
         kms_driven   -0.070029
         name         -0.069754
         company      -0.063714
         year          0.298133
         Price         1.000000
         Name: Price, dtype: float64
```

```
In [55]: # Plotting a barplot to see the relationship with  label in a better way

         car.drop('Price', axis=1).corrwith(car['Price']).plot(kind='bar', grid=True,figsize=(32,15),
                                                               title='Correlation with target')

         plt.show()
```



Achieving this score is a good sign that data science is really going to make a positive impact in the business decisions taken by management on the used car price prediction project

CONCLUSION

- Key Findings and Conclusions of the Study

Here's a general overview of how these inputs may be related to the price of a used car:

- Year: The year in which the car was manufactured is an important factor that can affect its price. As a car ages, it is likely to depreciate in value, although certain classic and luxury vehicles may increase in value over time.
- Price: The price of a used car is the most important factor in determining its value, and it is influenced by a number of other inputs. The price of a used car can be influenced by factors such as its age, condition, mileage, and features, as well as market demand and supply.
- Kilometers driven: The number of kilometers driven is a key factor that can affect the price of a used car. A car that has been driven fewer kilometers is generally considered to be in better condition than a car that has been driven more, and is therefore likely to command a higher price.
- Name of the car: The name of the car can also have an impact on its price, as certain car brands are more popular or prestigious than others. A luxury car from a well-known brand is likely to command a higher price than a car from a lesser-known brand, even if the two cars are similar in other ways.
- Manufacturing company: The manufacturing company can also have an impact on the price of a used car. Certain car manufacturers have a reputation for producing reliable and high-quality vehicles, which can command a higher price on the used car market.

- Fuel type: The fuel type can also have an impact on the price of a used car. Cars that run on alternative fuels, such as electric or hybrid vehicles, are becoming increasingly popular and may command a higher price on the used car market.
- The logic and output relationships between these inputs and the price of a used car can be determined through the use of machine learning algorithms, such as regression analysis or decision trees. These algorithms can analyze large amounts of data to identify patterns and relationships between the inputs and the price of a used car, and then use this information to make predictions about the price of new cars based on their inputs.

  Overall, the relationships between the inputs and the price of a used car are complex and can be influenced by many factors. By using data science and machine learning techniques, it is possible to build a model that can accurately predict the price of used cars based on these inputs.

  Studies from the data and the EDA done :

- We see that the year is left skewed and the price and kms driven are right skewed meaning the data is fairly new and old year is lesser and there are ouliers in price and kms driven.
- Maruthi has the highest number of cars in the dta set almost 80-90%
- Most of the cars are petrol and diesel and there are very very few lpg and electric cars
- In relationship graph between company and price , we see that benz and bmw cars have the highest range and volvo as well comes in very close

- In the same relation graph with year we see that 2016 to 2020 has the highest no of cars
- In the kms drive we see that the lower the kms driven the higher the price and the more it has the price decreases which means it has a negative relationship with price
- Fuel type as petrol and diesel have the highest more than 95% of data we see the highest range in them ‘

- Learning Outcomes of the Study in respect of Data Science

As mentioned in the previous part we have many features which have the most influence on the Target and having very impactful features it was really difficult to see prior to model building to come up with a good score or a good model , but since in Machine learning the computation is not statistical alone and the model actually learns the data as a whole , we are able to see such good scores. We see that in most situations we can go with Xgboost regressor as the model is not having overfitting or underfitting , it’s a strong model as well. If we are to use a substitute we can go for Random Forest regression as well as we saw it has a close score to Xgboost but the training was lesser, , But overall has a better and faster processing time so XGBoost regressor is the clear winner.

The study of a used car price prediction model in the context of data science can lead to several learning outcomes, including:

Understanding of data science concepts and techniques: Participants in the study of a used car price prediction model will gain a deeper understanding of various data science concepts and techniques, such as data cleaning, feature engineering, and model selection.

Understanding of machine learning algorithms: Participants will also learn about various machine learning algorithms, including regression and decision tree algorithms, and how these algorithms can be used to build a used car price prediction model.

Hands-on experience with data analysis and modeling: Participants will gain hands-on experience with data analysis and modeling by working with real-world datasets and building a used car price prediction model from scratch.

Knowledge of the used car market and industry: Participants will also gain an understanding of the used car market and industry, including the factors that influence used car prices and the challenges involved in predicting used car prices.

Understanding of the limitations and challenges of data-driven modeling: Participants will also gain an appreciation for the limitations and challenges of data-driven modeling, including issues related to data quality, model complexity, and subjectivity of car value.

Skills for data-driven decision making: By participating in the study of a used car price prediction model, participants will develop skills for data-driven decision making, including the ability to collect and analyze data, build models, and make predictions based on data-driven insights.

Overall, the study of a used car price prediction model in the context of data science can be a valuable learning experience that provides participants with a range of data science and machine

learning skills, as well as an understanding of the used car market and the challenges involved in predicting used car prices.

- Limitations of this work and Scope for Future Work

There are several limitations of a used car price prediction model that uses data science and machine learning techniques, including:

Data quality and availability: The accuracy of a used car price prediction model is largely dependent on the quality and availability of the data used to train the model. If the data used to train the model is incomplete, outdated, or otherwise unreliable, the predictions generated by the model may be inaccurate.

Model complexity: Building a used car price prediction model can be a complex and challenging task, as it requires the use of sophisticated algorithms and data analysis techniques. The model may also need to be constantly updated to reflect changes in the used car market and to account for new data inputs.

Subjectivity of car value: The value of a used car can be subjective and can vary depending on a number of factors, such as the buyer's preferences, the local market conditions, and the current state of the economy. This subjectivity can make it difficult to build a used car price prediction model that is accurate and reliable.

Despite these limitations, there is still significant scope for future work in the area of used car price prediction using data science and machine learning techniques. For example, there may be potential to improve the accuracy of the predictions generated by the model by incorporating more data inputs, such as the history of the car's maintenance and repairs, the local weather conditions, and the availability of similar vehicles in the local market.

Another area of future work may be to explore the use of alternative machine learning algorithms or data analysis techniques, such as deep learning or ensemble methods, to build a more accurate and reliable used car price prediction model.

Finally, there may be potential to use the used car price prediction model to develop new applications and services, such as a mobile app that provides used car price estimates based on the inputs entered by the user.

Overall, the field of used car price prediction using data science and machine learning is an active and rapidly evolving area of research, and there is significant potential for future work in this area to improve the accuracy and reliability of used car price predictions.