**Model creation to predict the price of used cars**

using machine learning

Problem Statement:

Predict price of used car based on various factors like manufacturer, year, model using Linear regression algorithm for designing our model.

Objective:

1. To get a better understanding of various learning algorithms and perform analysis to design a model for predicting used car prices.

2. Implement linear regression on the dataset to train our model for predicting the prices.

3. Use ROC and AOC to understand and visualize the data clearly.

4. A well-designed model will help users predict the price of their car and help them make knowledgeable decisions when they decide to buy or sell a car.

Application of our Project:

Individual use:

Many people have been interested in the used automobile market at some point in their lives because they wanted to sell or acquire a used vehicle. It's a great mistake to pay too much or sell for less than the market value in this process. As a result our application can help them

Car Dealers:

They are one of the most important target groups who may be interested in the study's findings. Used car dealers that have a better understanding of what makes a car desirable and what the most significant qualities are for a used automobile will be able to apply this knowledge and provide better service.

Workflow:

Acquire Dataset from Kaggle

Clean the Dataset (Remove Null values, extra columns

Split Dataset into training(85%) and testing(15%)

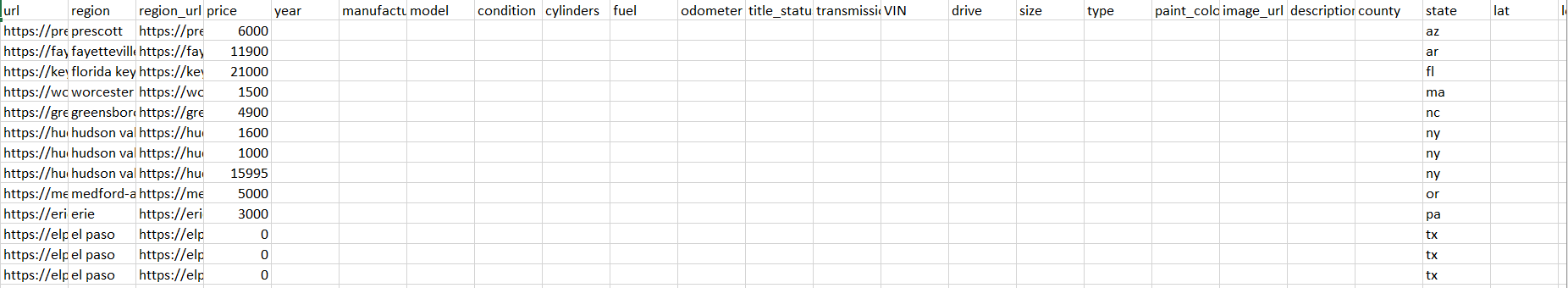
Develop the Model

Run the model after fitting data

Compare models based on their accuracy and generate results

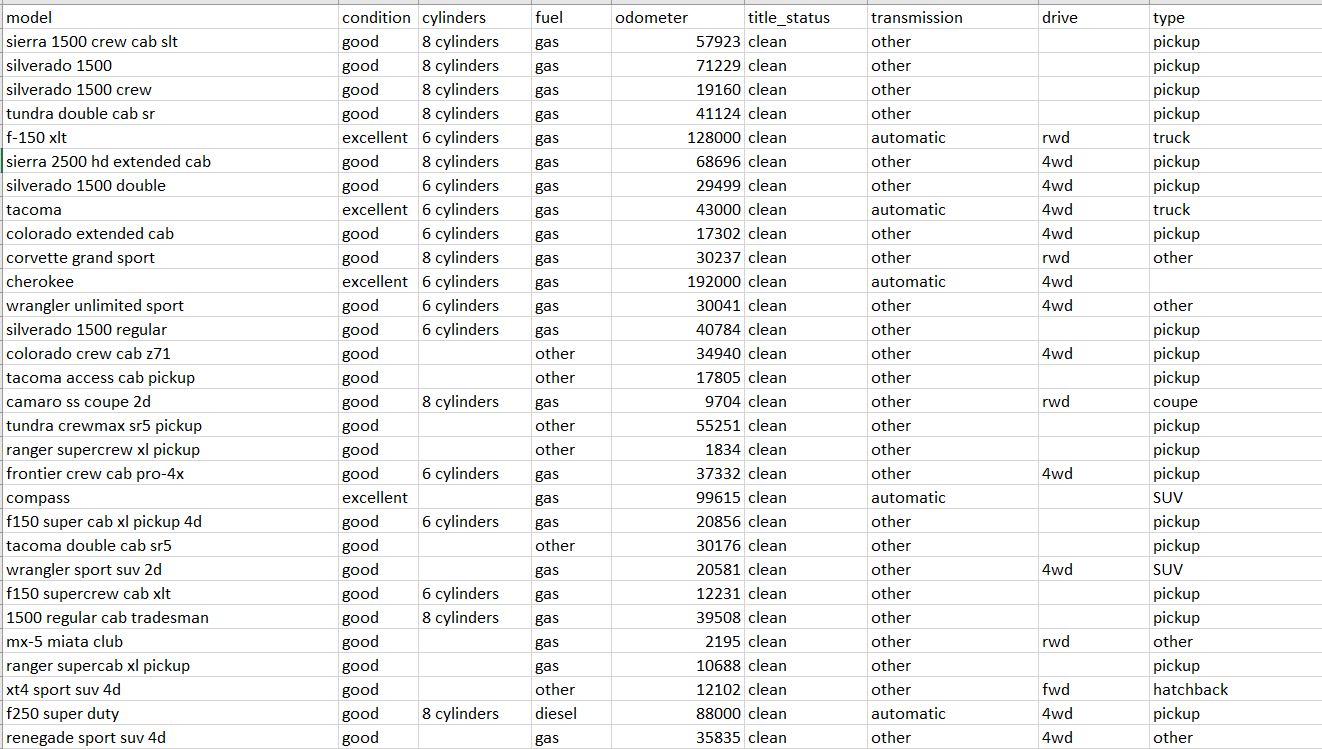
Dataset Description:

We acquired the dataset from Kaggle. The dataset initially had more than 400,000 rows and has the following attributes: url, region, region\_url, price, year, manufacturer, model, condition, cylinders, fuel, odometer, title\_status, transmission, VIN, drive, size, type, paint\_color,image\_url, description, country, state, lat, long,posting\_date.



The data had to be processed before training our model as it contained lots of entries with NULL, Nan or blank values. During processing we also removed columns like url, color lat, long etc. which served us no purpose, and then we removed the entries with null or blank value. We also checked if there were any duplicates. After processing the data, we had just over 250,000 rows of data.

Sample data post processing of raw data:



Description:

Model: Describes the model of the car.

Condition: Describes the condition the car is in. It can be good, excellent etc depending upon the performance of the car.

Cylinder, Transmission, Drive, Type and Fuel: Features of the car (which remain unchanged irrespective of whether it is new or old, but these are necessary details of a car).

Odometer: Describes the distance a car has traveled by the date of posting.

Libraries used for the project: We used python as our programming language for this project and these are the libraries that we used -

* Pandas - Used for manipulating data and analyzing it.
* Numpy - Provide support to use large multi-dimensional arrays and matrices along with the support to perform mathematical operations on these data structures.
* Seaborn - This library is used for data visualization.
* Matplotlib - It is mainly used for plotting graphs.

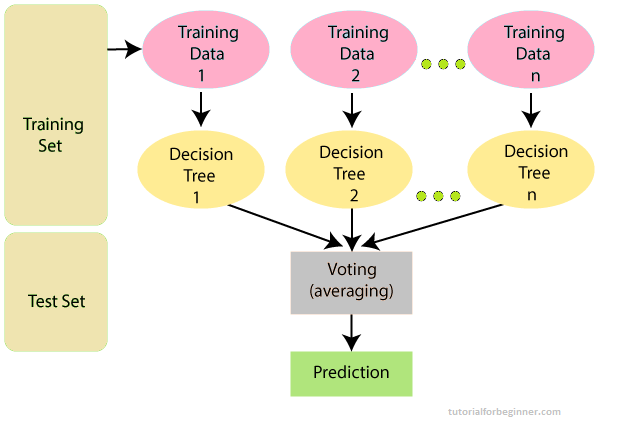
**Random Forest:**

Random forests, also known as random decision forests, are an ensemble learning approach for classification, regression, and other problems that works by training a large number of decision trees.The mean or average forecast of the individual trees is returned for regression.Random forests generally outperformdecision trees, but their accuracy is lower than gradient boosted trees.

A random forest algorithm consists of many decision trees. The ‘forest’ generated by the random forest algorithm is trained through bagging or bootstrap aggregating.In random forest algorithm establishing root nodes and segregating nodes is done randomly. Bagging is used to generate required predictions.

### Features of a Random Forest Algorithm

* It’s more accurate than the decision tree algorithm.
* It provides an effective way of handling missing data.
* It can produce a reasonable prediction without hyper-parameter tuning.
* It solves the issue of overfitting in decision trees.
* In every random forest tree, a subset of features is selected randomly at the node’s splitting point.



Random Forest Algorithm Working.

Step 1: Take N data points from the training set at random.

Step 2: For the data points you've picked, make decision trees.

Step 3: Choose a N for the number of decision trees you want to make.

Step 4: Repeat step 2 and step 3

Step 5: Find each decision tree's projections for new data points and assign them to the category with the highest votes.

**Linear Regression :**

Linear regression is mainly termed as a method commonly used for predictive analysis and modeling. It’s a model whose representation is a linear equation that combines specific input values for which solution is the predicted output value for given input values. It is basically used to predict the value of one variable using value of other variable. The variable which we are using to predict the value of another variable is termed an independent variable and the other termed as dependent variable. Linear Regression is mainly good for predicting the value of a given variable.

This technique finds a line that best “fits” the data and takes on the following form:

**ŷ = b0 + b1x**

where:

* **ŷ**: The estimated response value
* **b0**: The intercept of the regression line
* **b1**: The slope of the regression line

This equation can help us understand the relationship between the explanatory and response variable, and (assuming it’s statistically significant) it can be used to predict the value of a response variable given the value of the explanatory variable.

**K- Neighbors Regressor:**

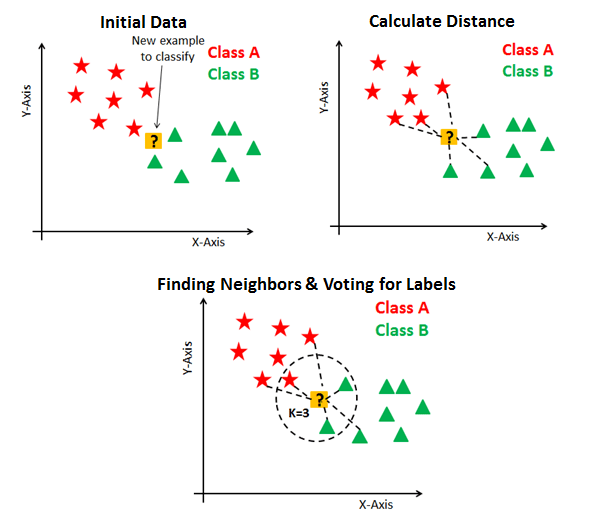
In K-Neighbour Regressor target is predicted by local interpolation of the targets associated of the nearest neighbors in the training set. It is an noo-parametric method. Euclidean or Manhatten function can be used to calculate the distance

Steps for K- Neighbors Regression:

Step 1: Choose the optimal value of K by inspecting the data

Step 2: Calculate euclidean distance of the target from the neighbour

Step 3: Calculate the target predicted value based on Neighbours

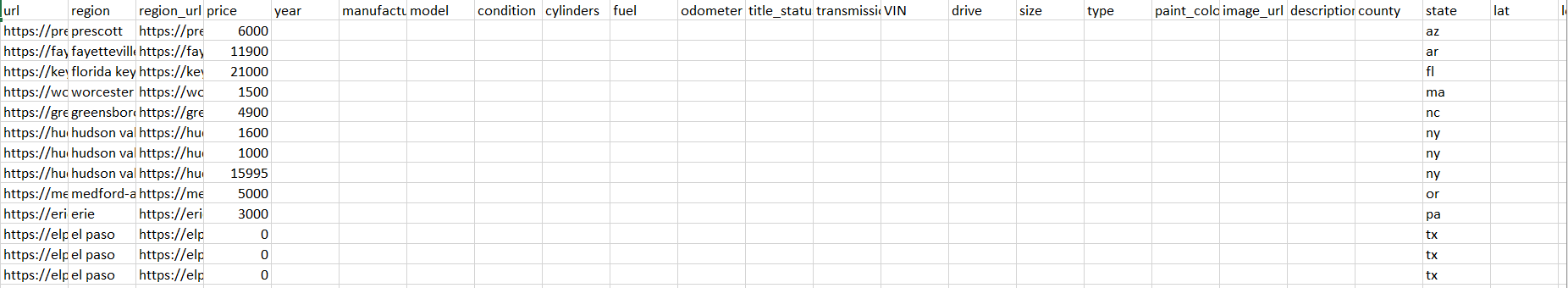


**Implementation:**

We followed the steps mentioned in the workflow earlier in the report. These were the steps involved.

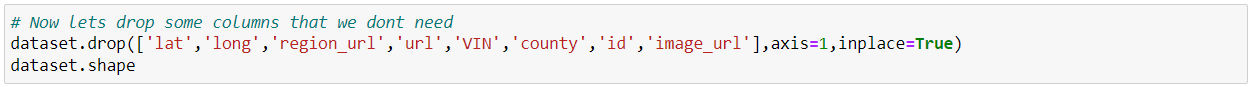
1. Define the problem statement i.e. what we wanted to achieve with this project and how to go about it.
2. Data gathering : Our data source was **Kaggle** and we got the dataset from this url : <https://www.kaggle.com/datasets/austinreese/craigslist-carstrucks-data>. The raw dataset had more than 400,000 records which consisted of blank, Null or Nan values too.

Snippet of the raw data.



1. Processing of the data: We had to remove multiple columns from our dataset as they served no useful purpose and occupied unnecessary space. We also filtered and removed duplicate, null or empty records. The whole task was done through python.

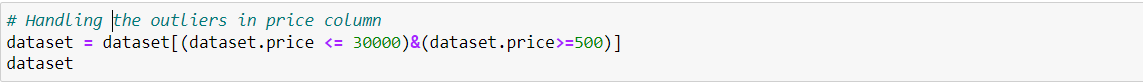
Snippets of the code used for processing:



Dropping unnecessary columns



Removing the duplicates

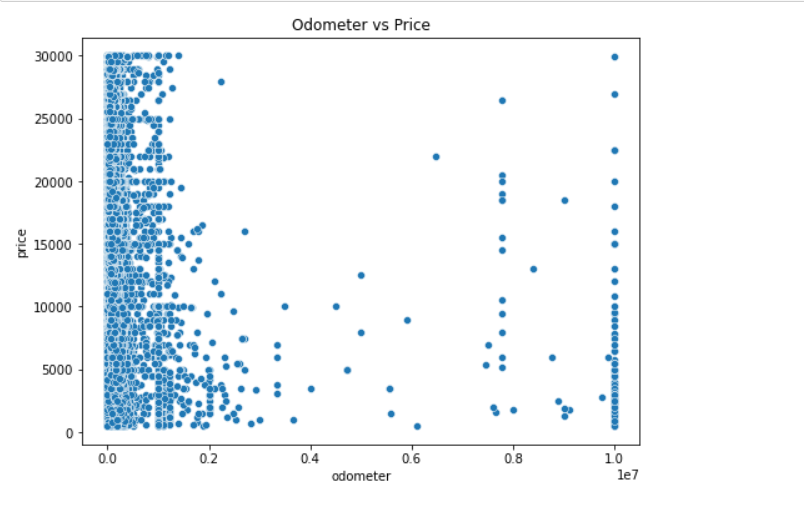


We decided to remove the outliers based on the price of the car so that we don’t have any car with price under 500 and above 30000.

1. Preliminary analysis: Before we started to train our model, we did some preliminary analysis to understand the trend a little better.

We plotted a scatter plot graph to understand the relation between odometer and the price of the car.

Result before processing the data:

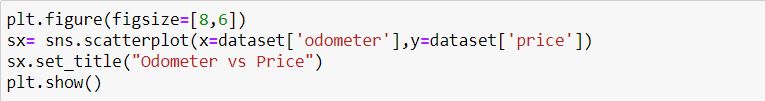


Result after processing the data:



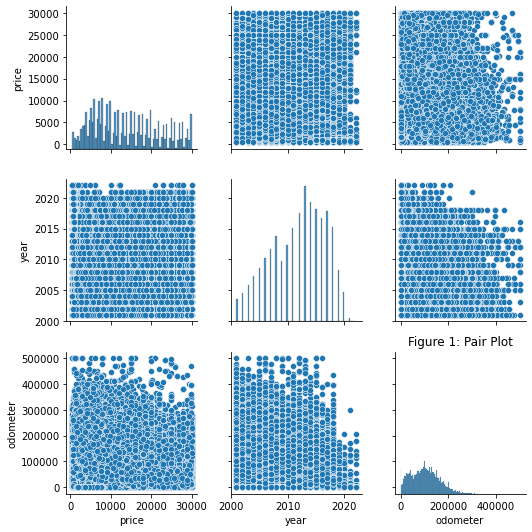
We got this result after we eliminated the outlier. For plotting this graph we used the python libraries.

Snippet of the code to plot the graph:

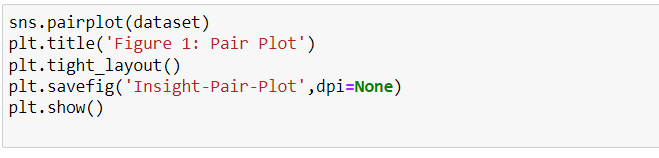


We also did a Pair plot for the dataset to compare multiple attributes with each other.

Pair plot after processing:

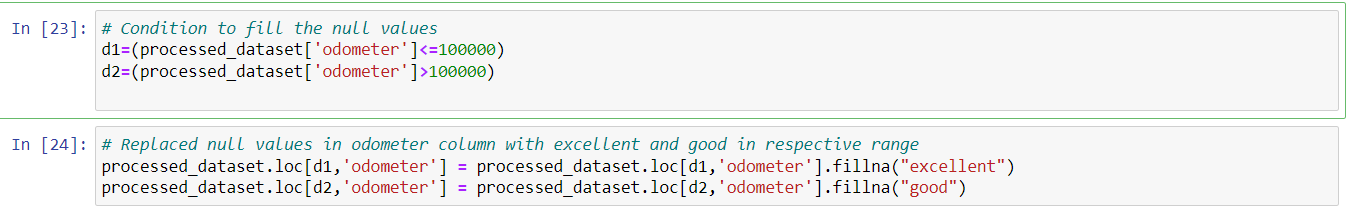


Snippet of code:

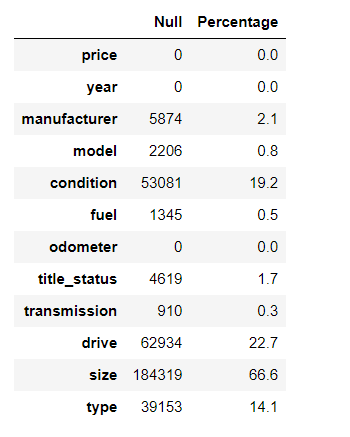


1. Filling in null values: We didn’t completely remove all the records with null value and instead decided to choose an average value for them based on other records. For e.g, If there were any values in the “condition” column with null values, we decided that if the odometer value was less than 100,00 then we will consider the car condition as “Excellent” otherwise it will be “good”.

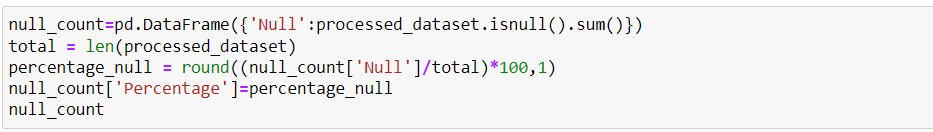
Snippet of the code to perform this task:



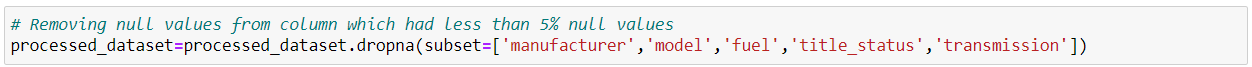
1. After this, we decided to do another check to see how many records were null for every column to get a better idea about the consistency of our dataset. This was the result:



Snippet of code to achieve this:

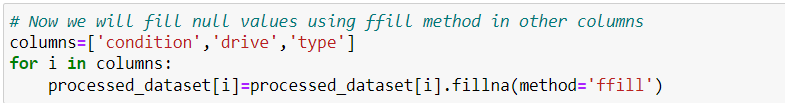


As seen in the above screenshot, column ‘size’ had the maximum number of null values so we decided to remove it from the dataset and for columns which had less than 5% null values, we removed the records for those columns with null values.

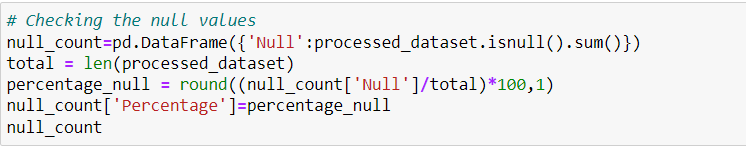




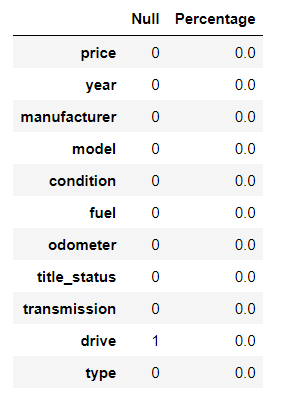
For the remaining columns, we used the **ffil** method from **pandas** library. **ffil** replaces the null value in a cell with the value in the cell above it. This way we eliminate all the null values from our dataset and run the check again to see if it still has any null values.



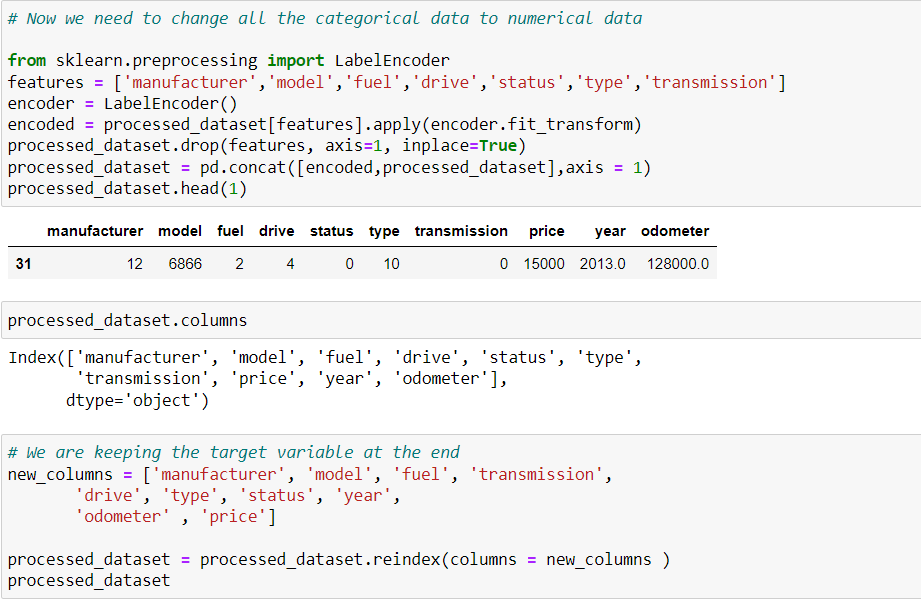
Checking for null values:



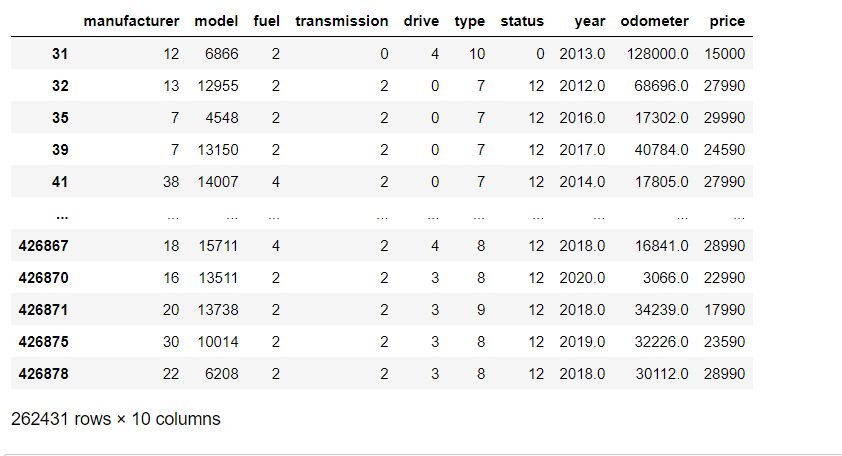
Result:



1. Now that we have eliminated all null values, the final step is to convert all categorical data like manufacturer, model, fuel etc. to numerical data. We used the library sklearn for this task.

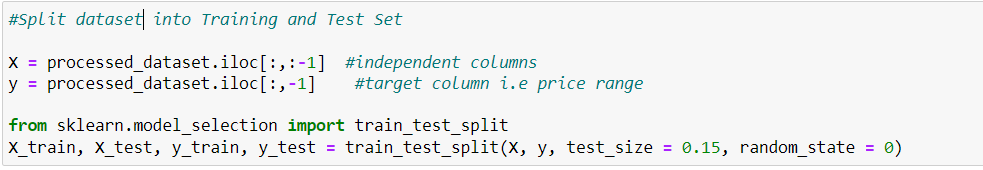


Result:

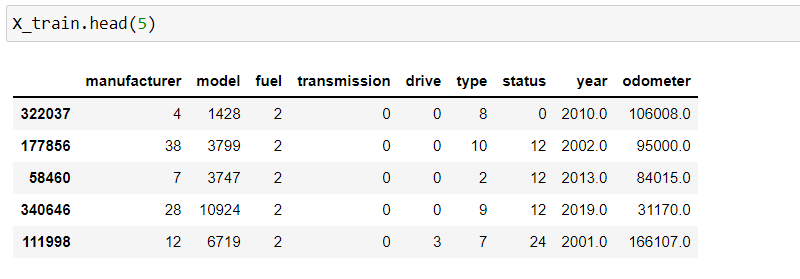


After this step, we are done with cleaning our dataset and now we start training our model.

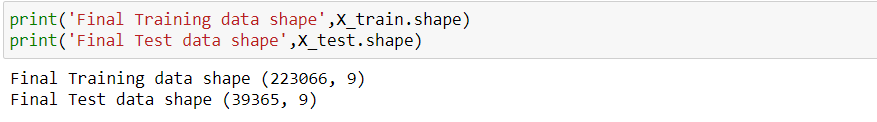
1. To train and test our model we have to split our dataset into training and test sets. We decided to go with an 85-15 split with 85% dataset to be used as training set and 15% will be used for testing our model.



Checking if training dataset is good by printing the first 5 rows:



We also confirmed that our split was good by checking the number of records in both, training and test, set.

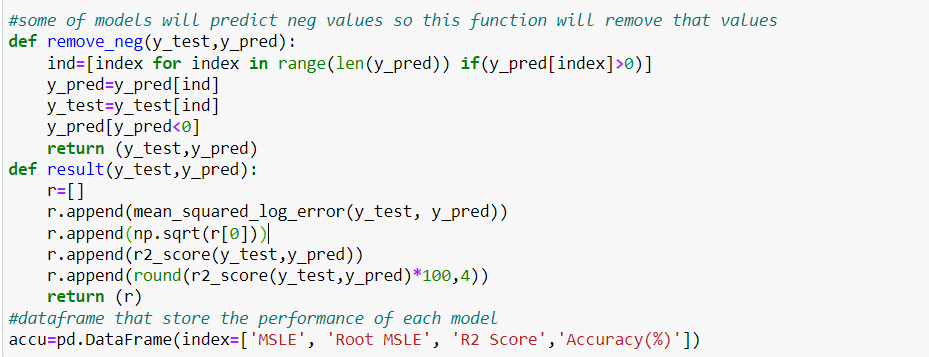


1. Next we implement 3 algorithms : Linear Regression, KNN and Random forest using the same method to compare their performance and note the observation.

Some of the common methods used in all three classification:



Defining dataset with 85% training and 15% test sets.

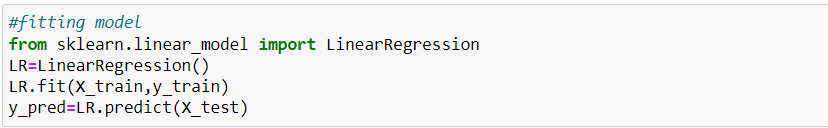


Dataframe to store the accuracy of the models.

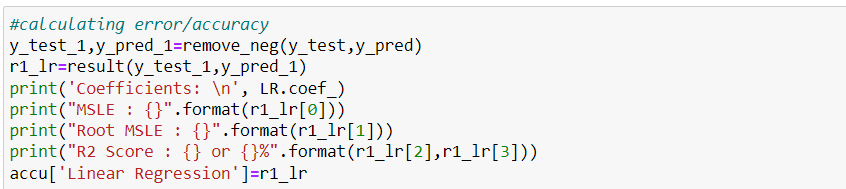
MSLE - Mean Squared Log Error

1. Using Linear Regression: We first implemented linear regression on our dataset. The following screenshot has the implementation and the result of the classification technique.

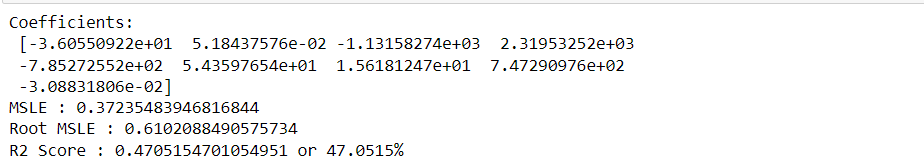
Implementing regression



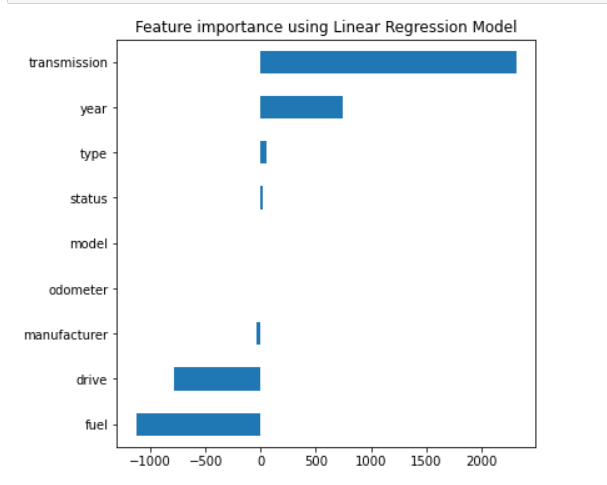
Calculating Accuracy



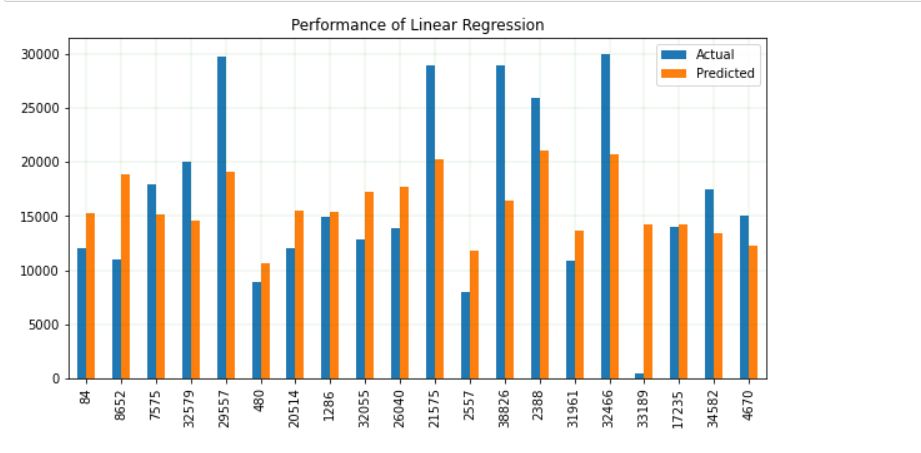
Result.



Importance of feature in the dataset:



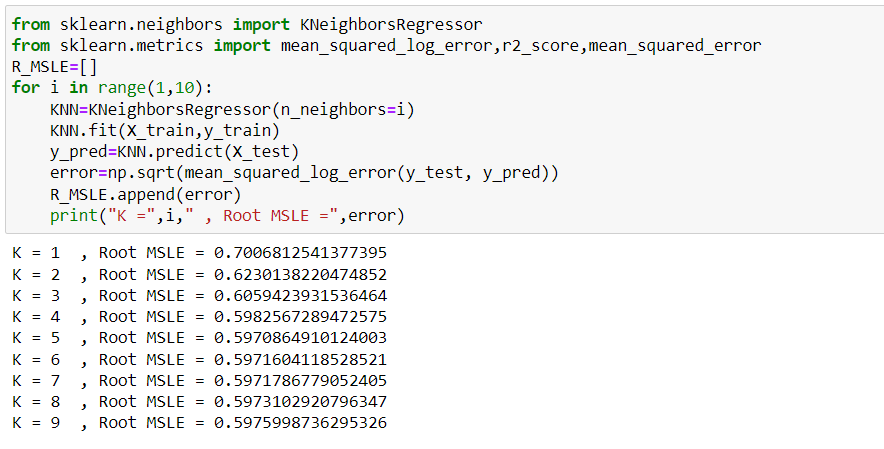
Graph to visualize Actual cost vs Predicted Cost:



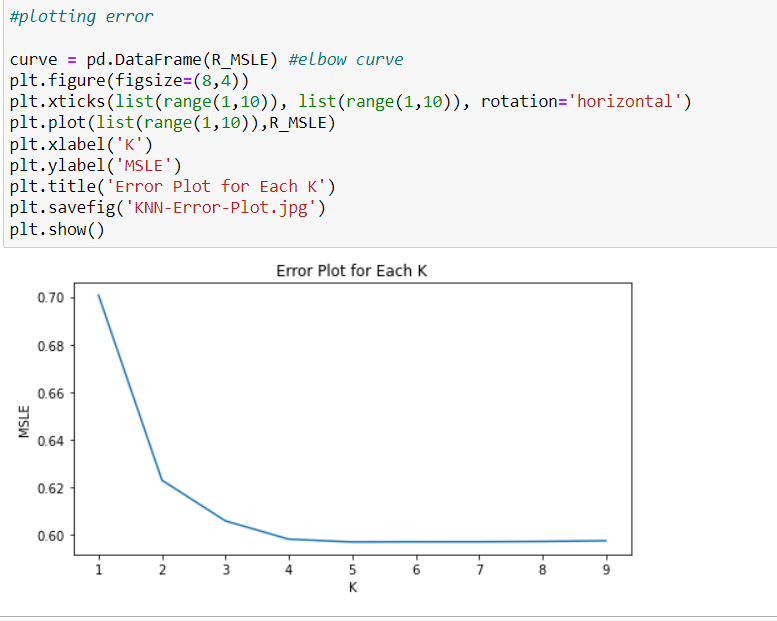
Score of Linear regression is approximately 49%.

1. Using K-nearest neighbor algorithm on our dataset:

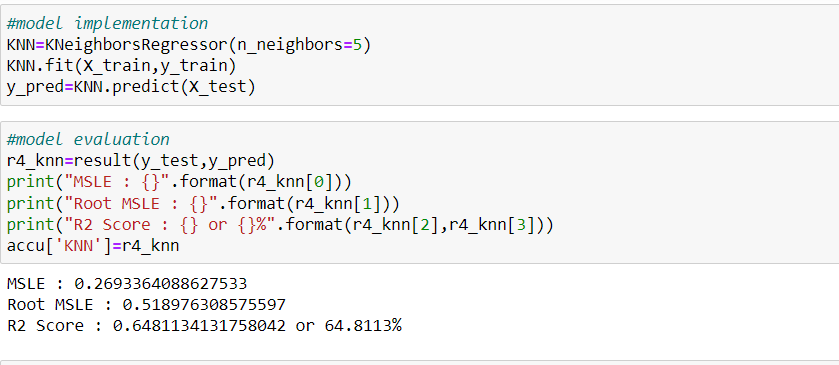
Implementation and Root Mean Squared Log Error :



Plotting the error graph for each ‘K’



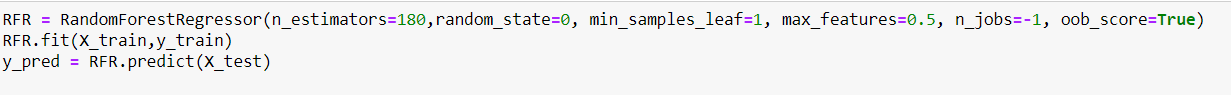
Model Implementation and Result:



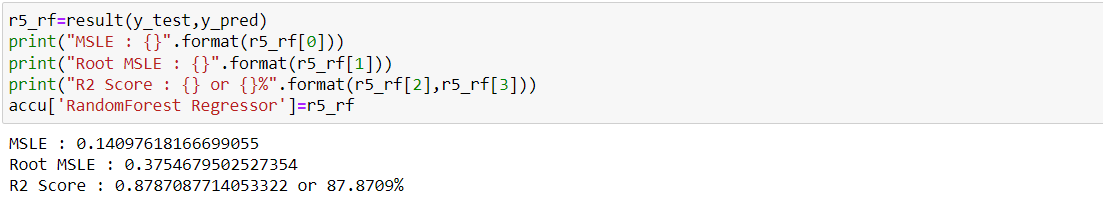
Score of KNN is approximately 65%.

1. We will now use Random forest classification on our dataset.

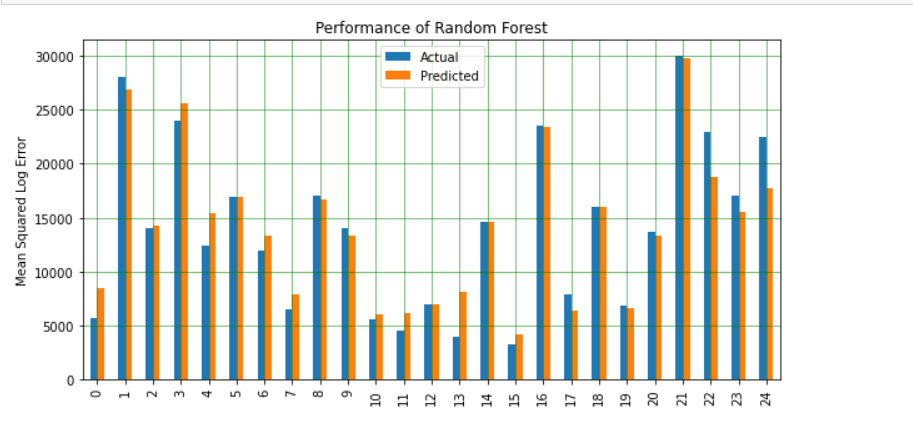
Implementation:



Result:



We also plotted a graph to visualize the performance of Random forest by comparing the actual cost versus predicted cost.



Score of Random Forest was approximately 88%.

Conclusion :

With this research, we created 3 models that can forecast the price of used automobiles based on a set of characteristics with an accuracy of 47.05,64.81,87.87 percent using Linear regression, K-nearest neighbor, Random forest classification. When it comes to estimating the worth of a car and, more crucially, the fundamental aspects that affect its cost, this knowledge may be quite useful for both companies and customers.

So we say Random forest is the best algorithm to find the price of used cars which has accuracy of approximately 88%.

References :

Austin Reese,

https://www.kaggle.com/datasets/austinreese/craigslist-carstrucks-data?resource=download

Josem,

https://github.com/josem-gp/Used-Car-Prices/blob/main/Used%20vehicles.ipynb

Shubham Jain,

https://shubh17121996.medium.com/used-car-price-prediction-using-supervised-machine-learning-ea9dace76686

Kshitij Kumbar, Pranav Gadre and Varun Nayak,

http://cs229.stanford.edu/proj2019aut/data/assignment\_308832\_raw/26612934.pdf

Abdulla AlShared ,

https://scholarworks.rit.edu/cgi/viewcontent.cgi?article=12220&context=theses

12-2021