Kaggle Project –

1)<https://www.kaggle.com/nehalbirla/vehicle-dataset-from-cardekho>

**Problem Statement**: What should be the car selling price that is used?.The dataset contains information about used cars. It can be implemented using linear regressions using Machine Learning**.(Why random forest and not regression algos.Random forest can be used for classification and regression problems.)**

1)Create a new environment:(Put libraries belong to that project only)

**conda create -n carprediction scipy**

**activate carprediction**

**cd C:\Users\deeps\OneDrive\Desktop\Ms\projects-sem1\datasets**

**conda install ipykernel**

**pip install jupyter**

**jupyter notebook**

**pip install pandas**

**import pandas as pd**

**df=pd.read\_csv('car data.csv')**

**df.head()**

2)Here we know that Selling\_Price is a dependent /target variable. And also Fuel\_Type,Seller\_Type,Transmission,Owner are categorical variables need to find the unqiue values.

**print(df['Fuel\_Type'].unique()) # prints unique values in the column**

**print(df['Transmission'].unique()) # prints unique values in the column**

**print(df['Seller\_Type'].unique()) # prints unique values in the column**

**print(df['Owner'].unique()) # prints unique values in the column**

3)Calculate the missing values /null values.

#check missing or null values

**df.isnull().sum()**

no null values so proceed.

4)Normalization:

The given dataset have a feature called as ‘Year’. The car depreciation may also depend on this feature so we convert this into No.of years dor ease.

No\_Years=Current\_Year – Given\_Year

\*We can skip the car names as it doesn’t play a major role in prediction.

**final\_dataset= df[[ 'Year', 'Selling\_Price', 'Present\_Price', 'Kms\_Driven',**

**'Fuel\_Type', 'Seller\_Type', 'Transmission', 'Owner']]**

**\***We can create the new feature Current\_Year in the final\_dataset .

**final\_dataset['Current\_Year']=2020**

**final\_dataset.head()**

**final\_dataset['no\_year']= final\_dataset['Current\_Year']- final\_dataset['Year']**

**final\_dataset.drop(['Year'],axis=1,inplace=True**) #axis=0/1 0 means labels and 1 means columns . inplace=True changes in original datatset and doesn’t return a copy

**final\_dataset.drop(['Current\_Year'],axis=1,inplace=True**)

**final\_dataset.head()**

5)Convert Categorical Features to 0’s and 1’s.

**final\_dataset=pd.get\_dummies(final\_dataset,drop\_first=True)** #drop\_first will delete the extra column(Original columns with categories)when converting categorical to dummy/indicator variables.

6)Find the Correlation of the columns-

**final\_dataset.corr()**

**pip install seaborn**

**import seaborn as sns**

**sns.pairplot(final\_dataset) #** The simplest invocation uses scatterplot() for each pairing of the variables and histplot() for the marginal plots along the diagonal**.**It plots pairwise

\*Create a Heat map and pairplot is not so clear.

**import matplotlib.pyplot as plt**

**%matplotlib inline** #When using the 'inline' backend, your matplotlib graphs will be included in your notebook, next to the code.This is called magic function as it allows to write code in command line style syntax.

**corrmat =final\_dataset.corr()**

**top\_corr\_features=corrmat.index**

**plt.figure(figsize=(20,20))**

**#Plot the heat map**

**sns.heatmap(final\_dataset[top\_corr\_features].corr(),annot=True,cmap="RdYlGn")** #annot=True means that the values should be present in the square brackets on the heat map .cmap specifies the color language of the heatmap.corr() uses Pearson Coreelation.Dark green means highly positively correlated and red means hghly negatively correated. If close to 1 or -1 then highly correlated based on positive and negative correlation.

**X=final\_dataset.iloc[:,1:]** #removing selling price as it is a depedent feature for new variable X.1: means leaving first everything.

**Y=final\_dataset.iloc[:,0]** #keeping only selling price

X->Independent feature

Y->Dependent Feature

#Feature importance (Tells which features are important and which are not)

**from sklearn.ensemble import ExtraTreesRegressor** #it is based on decision trees

**model=ExtraTreesRegressor()**

**model.fit(X,Y)**

**print(model.feature\_importances\_)** #Greater the value has higher importance .Eg 1,0.5,0.25 . 1st feature has highest importance compared to last feature with least importance of 0.25. map these with indices of X values.

\*plot graph for better visualization

**feat\_importances= pd.Series(model.feature\_importances\_,index=X.columns)** #make importance features values in x axis and convert to series.

**feat\_importances.nlargest(5).plot(kind='bar'**)#nlargest(5) plots the top features only

**plt.show()** #Based on the plot we can clearly see Present\_Price has the highest importance and the owner has the lowest importance.

\*Splitting the Dependent and Independent variables into train and test split:

#Split into training and testing data in order to test the accuracy of the model.

**from sklearn.model\_selection import train\_test\_split**

**X\_train,X\_test,Y\_train,Y\_test = train\_test\_split(X,Y,test\_size=0.2)**

**X\_train.shape** #(240,8)

This project uses Random Forest Regressor to predict the value. The scaling of values is not required as it basically uses Decision Tree internally.

https://www.youtube.com/watch?v=nxFG5xdpDto

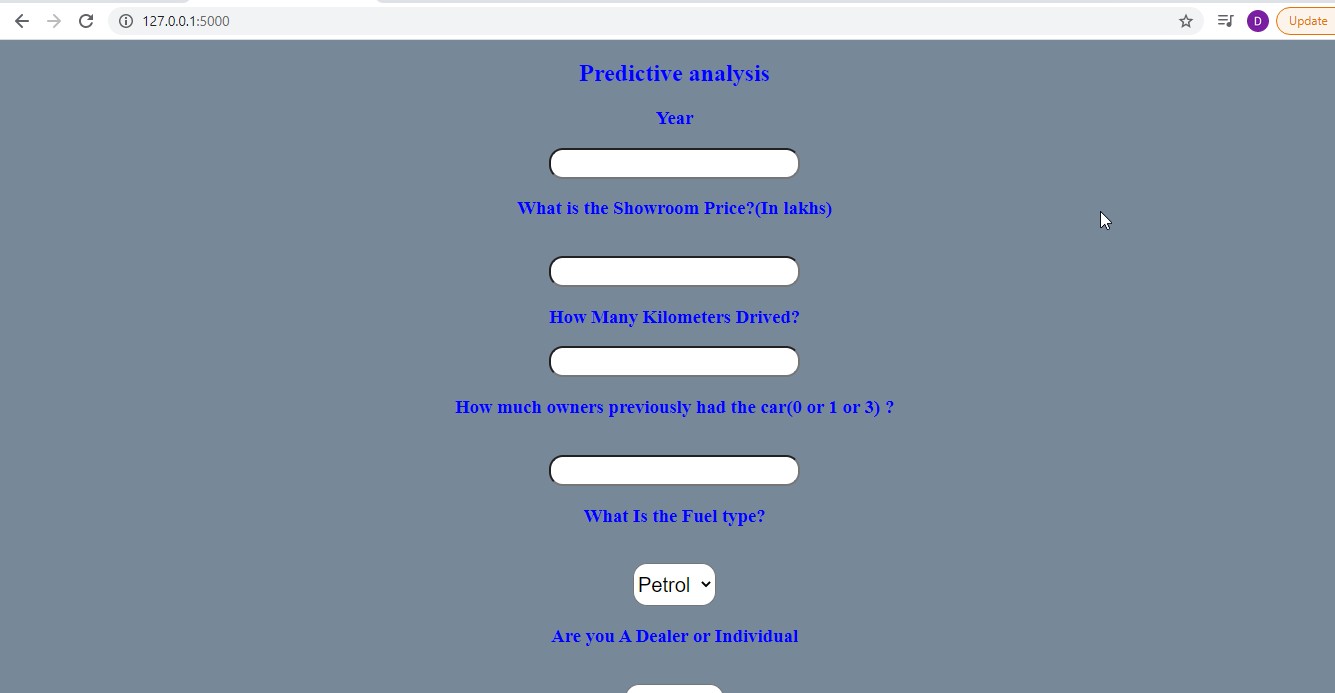
Details about Random Forest : **Random Forest** is a bagging technique .It is a classification algorithm.It is a collection of decision trees and based on Majority .It mostly works on all machine learning problems. In sklearn it finds the average of the decision trees for random forest. Classifier uses majority vote and Regressor uses average of the DT. It has low bias and high variance due to the sampling.

**Conclusion:**

Y\_test is the actual values and predictor is the formed values , both the difference must be minimal that’s why we get Gaussian distribution. Then we can predict the selling price based on other parameters using random forest techniques.



Output :



Based on the parameters predicts the selling price.



\***Deployment steps after pickling:**

1)Open Anaconda Prompt

2) activate carprediction

3) cd C:\Users\deeps\OneDrive\Desktop\Ms\projects-sem1\datasets

4) pip freeze > requirements.txt

5) pip install flask

6) python app.py

7)Go to the ip address and paste on chrome,