**Machine Learning Algorithms**

Use google colaboratory or jupyter notebooks to perform machine learning algorithms use the link below to sign in and work on futher

<https://research.google.com/colaboratory/>

if not installed install them using

!pip install pandas

import the necessary libraries

import pandas as pd

import seaborn as sns

from sklearn.linear\_model import LinearRegression

import matplotlib.pyplot as plt

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

from sklearn import metrics

first download the necessary data set by using the following website

<https://www.kaggle.com/>

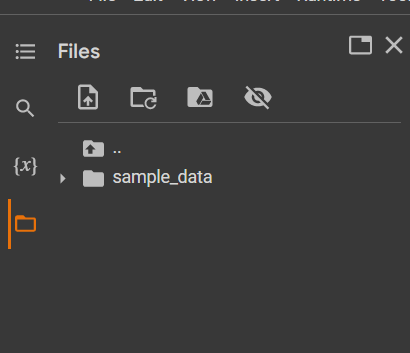
after downloading the csv files for processing, insert the data set in the google colaboratory by following steps

step 1:know the path where you have saved the csv file

step 2: go to google colaboratory then check for this in your left side



Then click the folder ,you will see a space like this then click on right of ur mouse u will get a pop select upload and update the csv file



To read a csv file ,use this command

data=pd.read\_csv("/content/car data.csv")

data(variable)

pd(denotes pandas as we have imported yearlier)

read\_csv (to read the csv file)

"/content/car data.csv" (data set path)

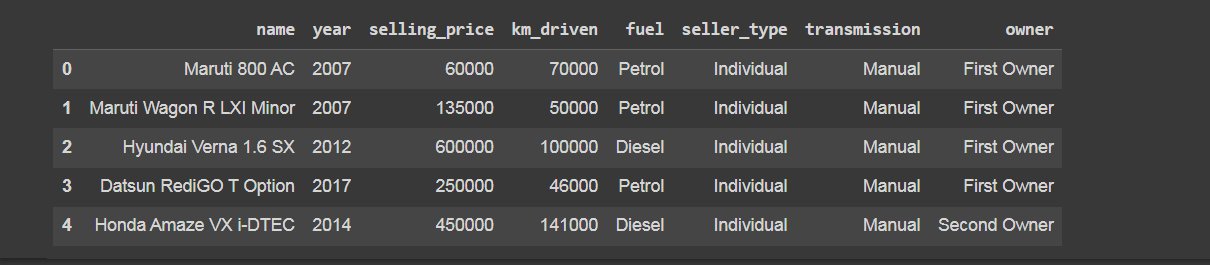
To fine the path ,select the file and search for kebab menu icon and select copy path to copy the actual path of the file

To run each cell press **ctrl+enter**

To print the 1st 4 datas then use

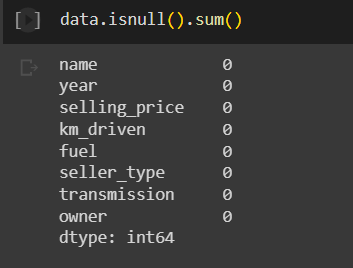
data.head()

#output

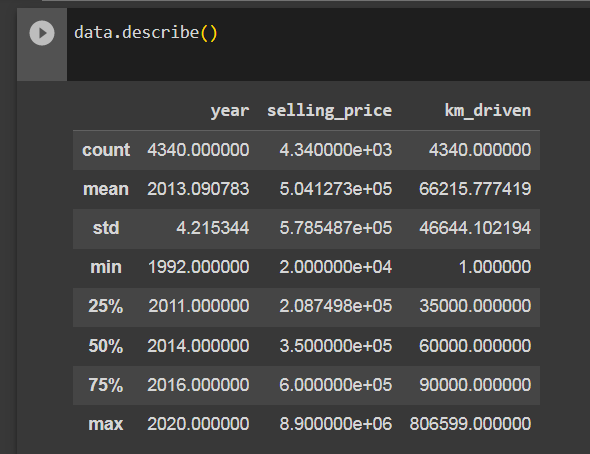


To find is there any null values exist,

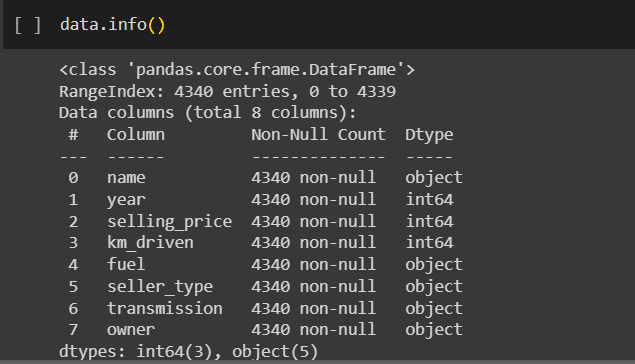
**data.isnull().sum()**



Here no null values exist.



info() function returns the data types in the dataset



Plot them

selling\_Price=data.groupby('year')['selling\_price'].sum()

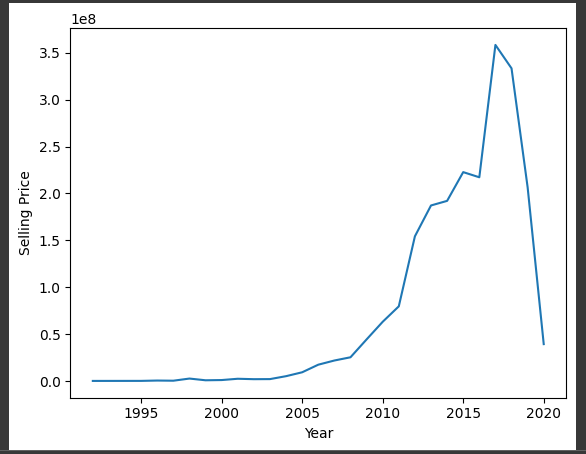
plt.plot(selling\_Price.index,selling\_Price.values)

plt.xlabel('Year')

plt.ylabel("Selling Price")

plt.show()

#output:



Linear regression:

Step 1: Importing the dataset  
Step 2: Data pre-processing  
Step 3: Splitting the test and train sets  
Step 4: Fitting the linear regression model to the training set  
Step 5: Predicting test results   
Step 6: Visualizing the test results

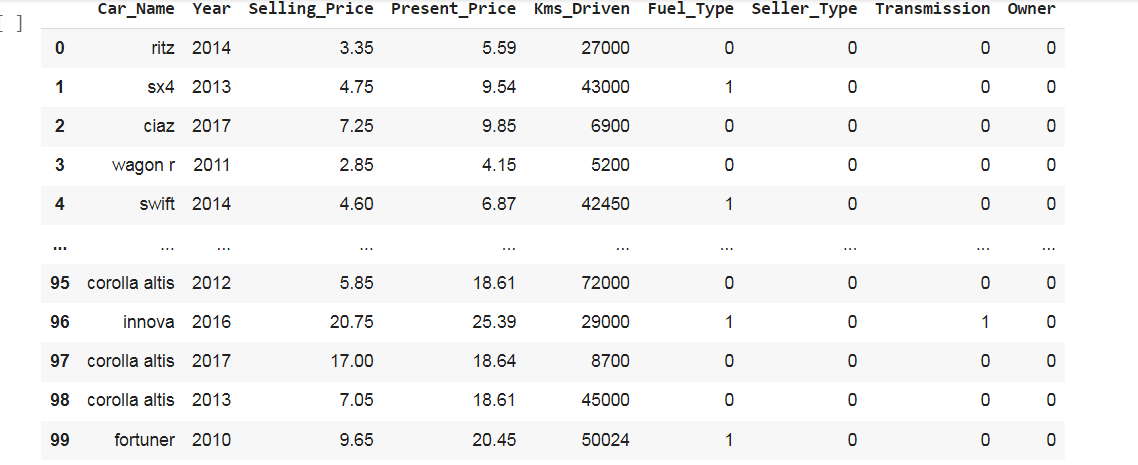
After viewing the data convert the sting values into int by using encoding methods

data.replace({'Fuel\_Type':{'Petrol':0,'Diesel':1,'CNG':2}},inplace=True)

data.replace({'Seller\_Type':{'Dealer':0,'Individual':1}},inplace=True)

data.replace({'Transmission':{'Manual':0,'Automatic':1,'Individual':2}},inplace=True)

this will replace the string values to numbers then view the output by printing the data set



Now split the data set,

X = data.drop(['Car\_Name','Selling\_Price'],axis=1)

Y = data['Selling\_Price']

x\_train,x\_test,y\_train,y\_test=train\_test\_split(X,Y,test\_size=0.2,random\_state=42)

now fit the model,

 model= LinearRegression()

 model.fit(x\_train,y\_train)

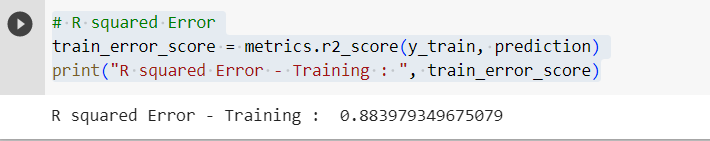
predict the test model,

prediction = model.predict(x\_train)

# R squared Error

train\_error\_score = metrics.r2\_score(y\_train, prediction)

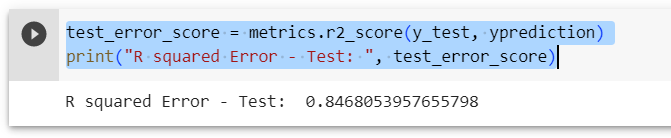
print("R squared Error - Training : ", train\_error\_score)



yprediction = model.predict(x\_test)

test\_error\_score = metrics.r2\_score(y\_test, yprediction)

print("R squared Error - Test: ", test\_error\_score)



Visualize the result,

plt.scatter(x\_train, y\_train, color="green")

plt.plot(x\_train, yprediction, color="red")

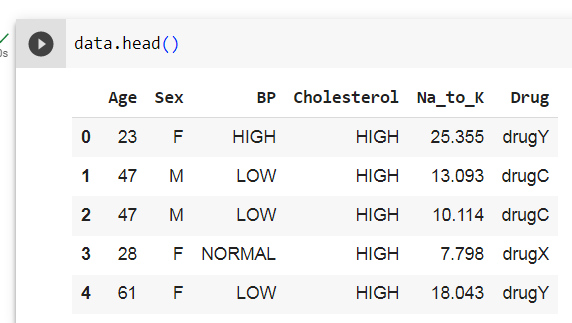
plt.title("used car dataset (Training Dataset)")

plt.show()

decision tree:

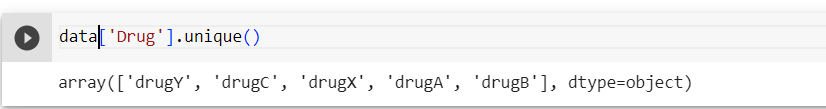
considering drugs data set to predict the drugs

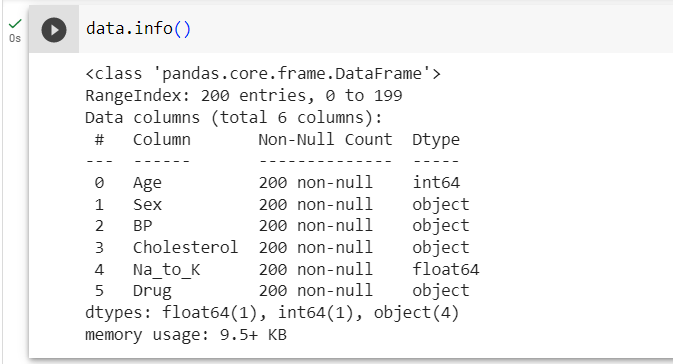
follow above procedure to import the dataset,then print 1st 4 data in the data set by using head()



To find the types of drugs involved we use unique() function

data['Drug'].unique()





* Convert Age, sex, BP, Cholesterol into 0,1 data by encoding

from sklearn.preprocessing import LabelEncoder

label = LabelEncoder()

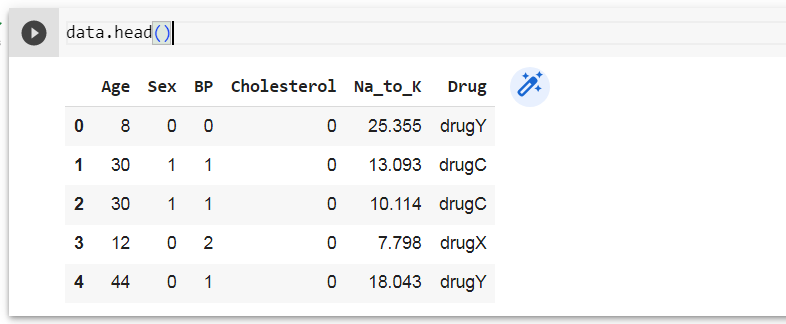
df['Age'] = label.fit\_transform(df['Age'])

df['Sex'] = label.fit\_transform(df['Sex'])

df['BP'] = label.fit\_transform(df['BP'])

df['Cholesterol'] = label.fit\_transform(df['Cholesterol'])

after encoding view the data,



x = data.iloc[:,:-1]

y = data.iloc[:,-1]

now split the data,

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

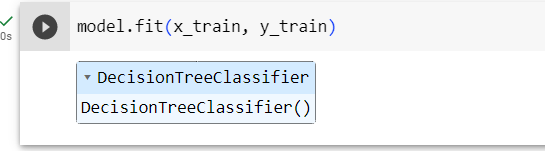
x\_train, x\_test, y\_train, y\_test = train\_test\_split(x,y,test\_size=0.2, random\_state=42)

Training the model using Decision Tree classifier

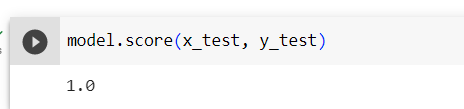
from sklearn.tree import DecisionTreeClassifier

model = DecisionTreeClassifier()

model.fit(x\_train, y\_train)



To check the accuracy ,we are getting 10% accuracy

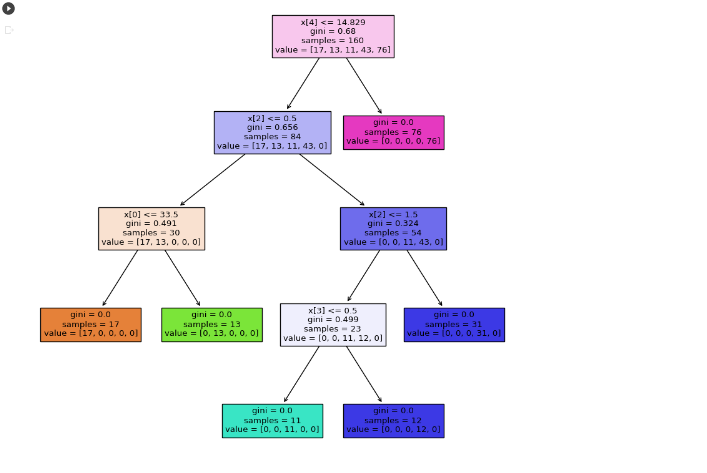


**Visualize the result**

from sklearn import tree

plt.figure(figsize=(10,10))

tree.plot\_tree(model, filled=True)

****

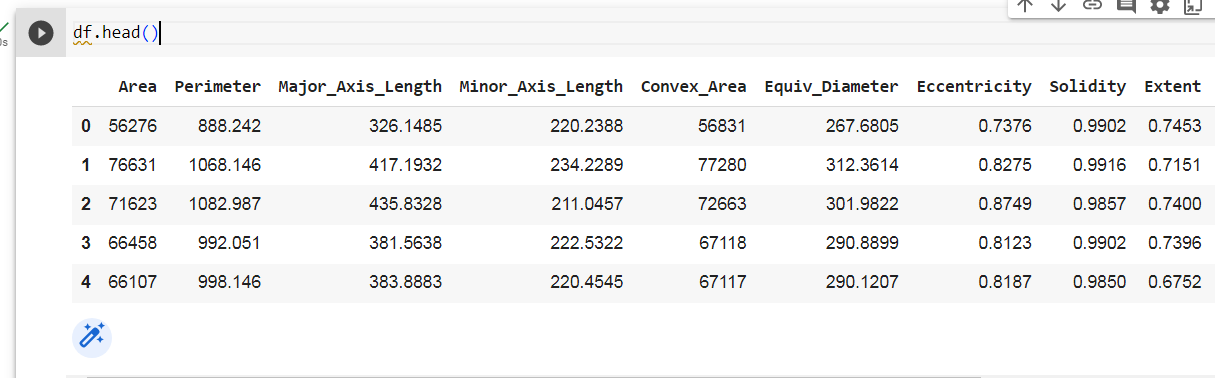
**Logistic regression:**

Considering pumpkin seed dataset from kaggle

Importing and reading the dataset

df=pd.read\_excel("/content/Pumpkin\_Seeds\_Dataset.xlsx")

df.head()



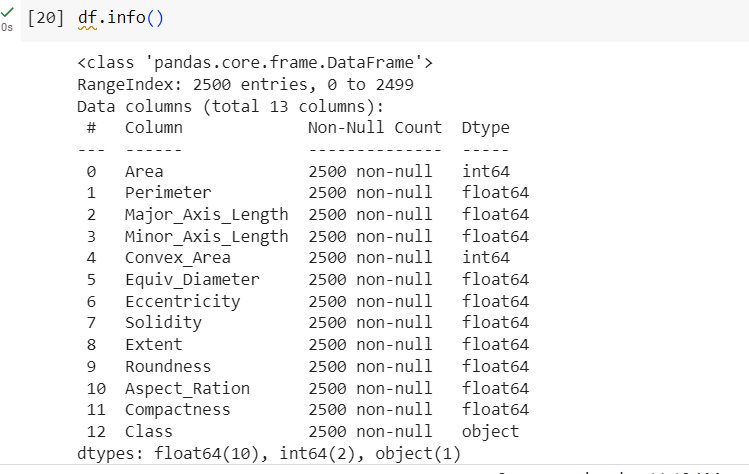
print(f"shape : {df.shape}")

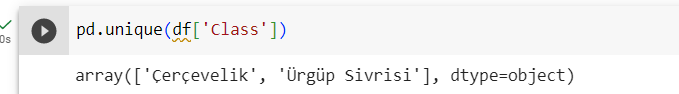
print(f"size  : {df.size}")

#output

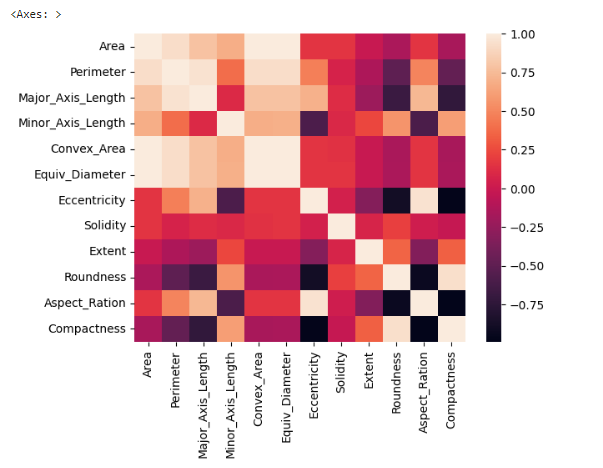
shape : (2500, 13)

size : 32500





sns.heatmap(df.drop(columns='Class').corr())



Split the data,

from sklearn.preprocessing import StandardScaler, LabelEncoder, PolynomialFeatures

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

x = np.array(df.drop(columns='Class'))

y = np.array(df['Class'])

# encode the target y

encoder = LabelEncoder()

encoder.fit(y)

y=encoder.transform(y)

y

x\_train,x\_test,y\_train,y\_test = train\_test\_split(x,y,train\_size=0.2)

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import r2\_score ,mean\_squared\_error, confusion\_matrix

lr = LogisticRegression( )

lr.fit(x\_train,y\_train)

lr.score(x\_test,y\_test)

Output

0.874

Visualize the model:

sns.heatmap(x\_train, annot=True, cmap='Blues')

plt.xlabel('Predicted labels')

plt.ylabel('True labels')

plt.show()