Car price Prediction

Installation

```
pip install numpy pandas scikit-learn pickle streamlit
```

Jupyter notebook

```
import pandas as pd(used of data manipulation and analysis, such as cleaning transforming)
import numpy as np(used for efficient numerical computations and handling large
multi-dimensional array and matrices)
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression

cars_data = pd.read_csv('Cardetails.csv')
cars_data.head()
cars_data.drop(columns='torque', inplace=True)
cars_data.shape

#Preprocessing
#Null Check
cars_data.isnull().sum()
cars_data.dropna(inplace=True)
cars_data.shape

#Duplicate Check
cars_data.duplicated().sum()
cars_data.drop_duplicates(inplace=True)
cars_data.shape
cars_data
cars_data.info()

#Data Analysis
for col in cars_data.columns:
    print('Unique value of ' + col)
    print(cars_data[col].unique)
    print('----------------------------')

def get_brand_name(car_name):
    car_name = str(car_name)
    car_name = car_name.split(' ')[0]
    return car_name.strip()

def clean_data(value):
    value = str(value)
    value = value.split(' ')[0]
    value = value.strip()
    if value == '':
        value = 0
    return float(value)
```

```
get_brand_name('Yamaha fzs 250')


cars_data['name'] = cars_data['name'].apply(get_brand_name)
cars_data['name'].unique()

cars_data['mileage'] = cars_data['mileage'].apply(clean_data)
cars_data['max_power'] = cars_data['max_power'].apply(clean_data)
cars_data['engine'] = cars_data['engine'].apply(clean_data)

for col in cars_data.columns:
    print('Unique value of ' + col)
    print(cars_data[col].unique())
    print('======================')

cars_data['name'] = cars_data['name'].replace(['Maruti', 'Skoda', 'Honda', 'Hyundai', 'Toyota',
'Ford', 'Renault',
       'Mahindra', 'Tata', 'Chevrolet', 'Datsun', 'Jeep', 'Mercedes-Benz',
       'Mitsubishi', 'Audi', 'Volkswagen', 'BMW', 'Nissan', 'Lexus',
       'Jaguar', 'Land', 'MG', 'Volvo', 'Daewoo', 'Kia', 'Fiat', 'Force',
       'Ambassador', 'Ashok', 'Isuzu', 'Opel'],

[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31])

cars_data['transmission'].unique()
cars_data['transmission'] = cars_data['transmission'].replace(['Manual', 'Automatic'],[1,2])

cars_data['seller_type'].unique()
cars_data['seller_type'] = cars_data['seller_type'].replace(['Individual', 'Dealer', 'Trustmark
Dealer'],[1,2,3])
cars_data.info()

cars_data['fuel'].unique()
cars_data['fuel'] = cars_data['fuel'].replace(['Diesel', 'Petrol', 'LPG', 'CNG'],[1,2,3,4])
cars_data.info()

cars_data
cars_data.reset_index(inplace=True)
cars_data

cars_data['owner'].unique()
cars_data['owner'] = cars_data['owner'].replace(['First Owner', 'Second Owner', 'Third
Owner',
       'Fourth & Above Owner', 'Test Drive Car'],[1,2,3,4,5])
cars_data

cars_data.drop(columns=['index'], inplace=True)
```

cars_data

input_data = cars_data.drop(columns=['selling_price'])
output_data = cars_data['selling_price']
x_train, x_test, y_train, y_test = train_test_split(input_data, output_data, test_size=0.2)(used to split your dataset into training and testing subsets. This is a crucial step in machine learning to ensure that your model is trained on one portion of the data and tested on another, separate portion.)

#model creation
model = LinearRegression()(Create an instance of the LinearRegression model)

#Train model
model.fit(x_train, y_train)(model(x_train, y_train) is used to train the linear regression model using the training data.)

predict = model.predict(x_test)(The model has already been trained on the training data (x_train and y_train) using model.fit(x_train, y_train).During training, the model learned the relationship between the features (independent variables) and the target variable (dependent variable))
predict

x_train.head(1)
input_data_model = pd.DataFrame([[9,2022,195000,1,1,1,2,21.12,1248.0,88.8,5.0]], columns=['name','year','km_driven','fuel','seller_type','transmission','owner','mileage','engine', 'max_power','seats'])

input_data_model
model.predict(input_data_model)

#Pickle
import pickle as pk(pickle used to serialise and save models to disk and later deserialize and load them for use without retraining.Serialization refers to the process of converting a Python object into a byte stream or a string representation that can be easily stored, transmitted over a network, or saved to disk.)
pk.dump(model,open('model.pkl','wb'))(The line pk.dump(model, open('model.pkl', 'wb')) saves the trained Linear Regression model (model) to a file named 'model.pkl' using Python's built-in pickle)

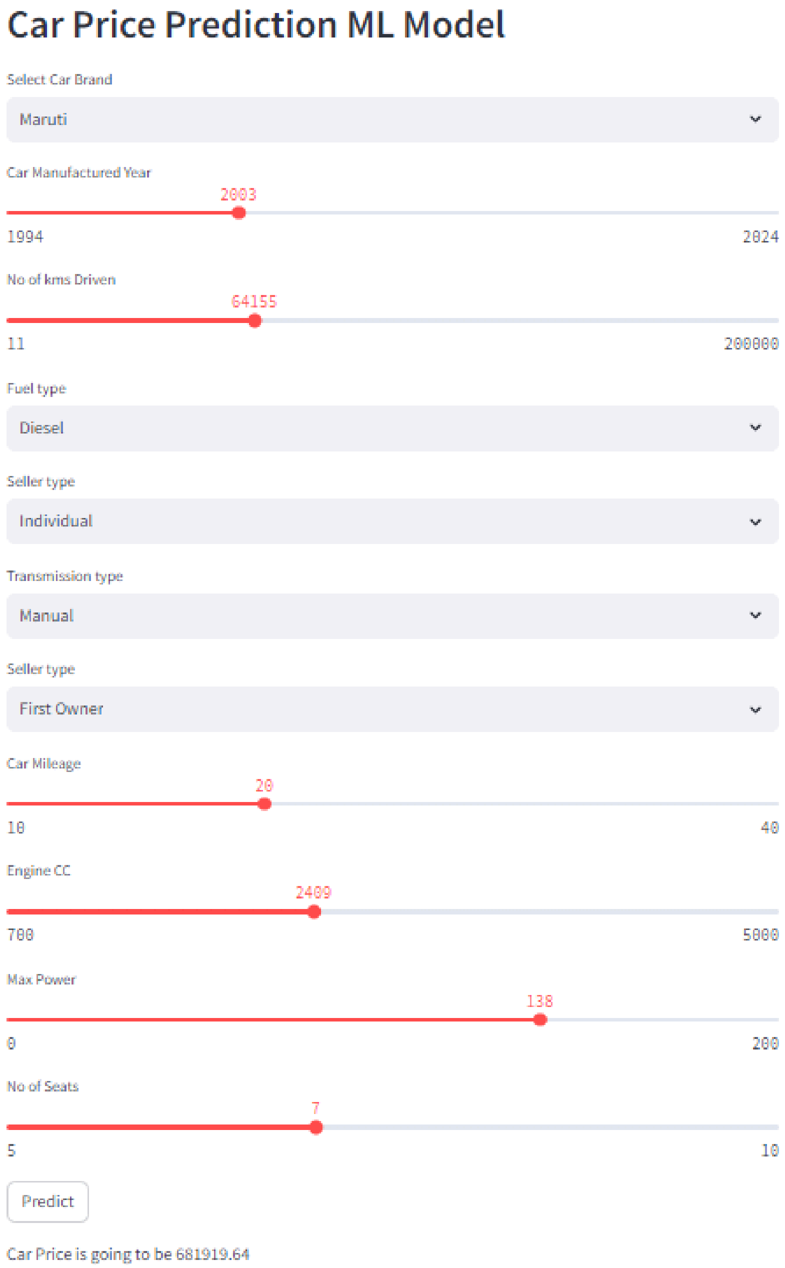
Vscode

```
import pandas as pd
import numpy as np
import pickle as pk
import streamlit as st
```

```
model = pk.load(open('model.pkl','rb'))

st.header('Car Price Prediction ML Model')

cars_data = pd.read_csv('Cardetails.csv')

def get_brand_name(car_name):
    car_name = car_name.split(' ')[0]
    return car_name.strip()
cars_data['name'] = cars_data['name'].apply(get_brand_name)

name = st.selectbox('Select Car Brand', cars_data['name'].unique())
year = st.slider('Car Manufactured Year', 1994,2024)
km_driven = st.slider('No of kms Driven', 11,200000)
fuel = st.selectbox('Fuel type', cars_data['fuel'].unique())
seller_type = st.selectbox('Seller  type', cars_data['seller_type'].unique())
transmission = st.selectbox('Transmission type', cars_data['transmission'].unique())
owner = st.selectbox('Seller  type', cars_data['owner'].unique())
mileage = st.slider('Car Mileage', 10,40)
engine = st.slider('Engine CC', 700,5000)
max_power = st.slider('Max Power', 0,200)
seats = st.slider('No of Seats', 5,10)


if st.button("Predict"):
    input_data_model = pd.DataFrame(

[[name,year,km_driven,fuel,seller_type,transmission,owner,mileage,engine,max_power,seat
s]],

columns=['name','year','km_driven','fuel','seller_type','transmission','owner','mileage','engine',
'max_power','seats'])

    input_data_model['owner'].replace(['First Owner', 'Second Owner', 'Third Owner',
       'Fourth & Above Owner', 'Test Drive Car'],
                           [1,2,3,4,5], inplace=True)
    input_data_model['fuel'].replace(['Diesel', 'Petrol', 'LPG', 'CNG'],[1,2,3,4], inplace=True)
    input_data_model['seller_type'].replace(['Individual', 'Dealer', 'Trustmark Dealer'],[1,2,3],
inplace=True)
    input_data_model['transmission'].replace(['Manual', 'Automatic'],[1,2], inplace=True)
    input_data_model['name'].replace(['Maruti', 'Skoda', 'Honda', 'Hyundai', 'Toyota', 'Ford',
'Renault',
       'Mahindra', 'Tata', 'Chevrolet', 'Datsun', 'Jeep', 'Mercedes-Benz',
       'Mitsubishi', 'Audi', 'Volkswagen', 'BMW', 'Nissan', 'Lexus',
       'Jaguar', 'Land', 'MG', 'Volvo', 'Daewoo', 'Kia', 'Fiat', 'Force',
       'Ambassador', 'Ashok', 'Isuzu', 'Opel'],

[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31]
```

```
                    ,inplace=True)

    car_price = model.predict(input_data_model)

    st.markdown('Car Price is going to be {:.2f}'.format(car_price[0]))
```

Browser



Car Price Prediction ML Model

Select Car Brand

Maruti

Car Manufactured Year

2003

1994 2024

No of kms Driven

64155

11 200000

Fuel type

Diesel

Seller type

Individual

Transmission type

Manual

Seller type

First Owner

Car Mileage

20

10 40

Engine CC

2409

700 5000

Max Power

138

0 200

No of Seats

7

5 10

Car Price is going to be 681919.64