Petrol price prediction

Using Machine Learning

Name: Kailash N Email:kailash31057@gmail.com Github: https://github.com/Nkailash7/Petrol-Price-Prediction-using-Machine-Learning/blob/main/petrol-prices.ipynb

ABSTRACT

In day today life Worldwide the most trending analysis is on Petrol price deviations along forecasting analysis required for everyday activities. Production varies in each states and countries which needs analysis that helps to reduce the demand in fuels and gases. The vital and global need for transportation is fuel needs and on time availability in each and every place. Global positioning system (GPS) updated satellite locations for vehicle navigation to increase the required availability of fuels. Machine Learning (ML) an accurate technique to consume the fuel usage based on the travel distance and prediction using consumption of petrol. We propose a prediction model using random forest algorithm for statistical analysis and attribute based on the dataset collected. We have collected a petrol price variation data from bank bazar and gathered the price variation based on highest and lowest changes. From 2019 to 2021 march petrol price as highest and lowest at the end of year and its performance mentioned as Rise, Decrease, Stable, Unstable as an attributes for consumption range. Using random forest algorithm 87% of accuracy achieved and data attributes splits the data according to the decision making points and avoid overfitting to the values which identifies the Euclidean distance according to the independent parameters which produces accuracy in proposed model.

# This dataset consists of information about country, their world share, daily oil consumption, etc.

Data set: https://www.kaggle.com/datasets/zusmani/petrolgas-prices-worldwide

# ACKNOWLEDGE

I am using this opportunity to express my gratitude to everyone who supported me throughout the course of my capstone project. I am thankful for their aspiring guidance, invaluably constructive criticism and friendly advice during the project work. I am sincerely grateful to them for sharing their truthful and illuminating views on a number of issues related to the project.

Further, I have fortunate to have Mr.Anbu Joel as my mentor. He has readily shared his immense knowledge in data analytics and guide me in a manner that the outcome resulted in enhancing my data skills.

I certify that the work done by me for conceptualizing and completing this

project is original and authentic.

Date: 2/7/2022 Name: KAILASH N

# **Table Of Content**

|  |  |  |
| --- | --- | --- |
| Chapter Number | Title | Page Number |
| 1. | Introduction |  |
| 2. | Background |  |
| 3. | Implementation Result |  |
| 4. | Performance Metrics |  |
| 5. | Conclusion |  |

INTRODUCTION

Good eating is essential to human health [1]. Natural products have been commonly used as foodstuffs

and can also be manufactured to fulfill market demand. Food attributes such as type, structure,

nutrients, and process types (natural products and refined food) are concerned with balanced diet

issues. It is a fact that individuals have different eating patterns from other areas. Knowing the

characteristics of foods (type, composition, nutrients and process types, etc.) The consistency and

protection of foods for customers worldwide is essential for the inspection [2]. A realistic demand in

everyday life is swift, precise, and automatic determination of food attributes. Modern techniques have

been commonly used to detect food characteristics, including electronic noses, computer vision,

spectroscopy and spectral imaging, and so on [3, 4]. A large amount of digital data relating to food

properties can be collected through such methods. Data analysis of these methods is important because

the enormous data volume includes a lot of repetitive and irrelevant material [5]. It is an urgent and

vital challenge to deal with such a vast volume of data and extract useful features from the acquired

data, and the complexity of bringing these methods into real-world use [6, 7]

Good eating is essential to human health [1]. Natural products have been commonly used as foodstuffs

and can also be manufactured to fulfill market demand. Food attributes such as type, structure,

nutrients, and process types (natural products and refined food) are concerned with balanced diet

issues. It is a fact that individuals have different eating patterns from other areas. Knowing the

characteristics of foods (type, composition, nutrients and process types, etc.) The consistency and

protection of foods for customers worldwide is essential for the inspection [2]. A realistic demand in

everyday life is swift, precise, and automatic determination of food attributes. Modern techniques have

been commonly used to detect food characteristics, including electronic noses, computer vision,

spectroscopy and spectral imaging, and so on [3, 4]. A large amount of digital data relating to food

properties can be collected through such methods. Data analysis of these methods is important because

the enormous data volume includes a lot of repetitive and irrelevant material [5]. It is an urgent and

vital challenge to deal with such a vast volume of data and extract useful features from the acquired

data, and the complexity of bringing these methods into real-world use [6, 7]

Good eating is essential to human health [1]. Natural products have been commonly used as foodstuffs

and can also be manufactured to fulfill market demand. Food attributes such as type, structure,

nutrients, and process types (natural products and refined food) are concerned with balanced diet

issues. It is a fact that individuals have different eating patterns from other areas. Knowing the

characteristics of foods (type, composition, nutrients and process types, etc.) The consistency and

protection of foods for customers worldwide is essential for the inspection [2]. A realistic demand in

everyday life is swift, precise, and automatic determination of food attributes. Modern techniques have

been commonly used to detect food characteristics, including electronic noses, computer vision,

spectroscopy and spectral imaging, and so on [3, 4]. A large amount of digital data relating to food

properties can be collected through such methods. Data analysis of these methods is important because

the enormous data volume includes a lot of repetitive and irrelevant material [5]. It is an urgent and

vital challenge to deal with such a vast volume of data and extract useful features from the acquired

data, and the complexity of bringing these methods into real-world use [6, 7]

Good eating is essential to human health [1]. Natural products have been commonly used as foodstuffs

and can also be manufactured to fulfill market demand. Food attributes such as type, structure,

nutrients, and process types (natural products and refined food) are concerned with balanced diet

issues. It is a fact that individuals have different eating patterns from other areas. Knowing the

characteristics of foods (type, composition, nutrients and process types, etc.) The consistency and

protection of foods for customers worldwide is essential for the inspection [2]. A realistic demand in

everyday life is swift, precise, and automatic determination of food attributes. Modern techniques have

been commonly used to detect food characteristics, including electronic noses, computer vision,

spectroscopy and spectral imaging, and so on [3, 4]. A large amount of digital data relating to food

properties can be collected through such methods. Data analysis of these methods is important because

the enormous data volume includes a lot of repetitive and irrelevant material [5]. It is an urgent and

vital challenge to deal with such a vast volume of data and extract useful features from the acquired

data, and the complexity of bringing these methods into real-world use [6, 7].

Several data analysis methods, such as partial least squares (PLS), artificial neural netw

As we are experiencing an unstable increase in prices of petrol

where the oil prices are dependent on the crude oil

prices of Dubai and Saudi-Arabia. The transportation will be

affected for change in prices. The petrol price of India

has also been taken into consideration for the accurate

prediction of prices. I have used various algorithms for

predicting the diesel price in India. The algorithms which

we have used are Random Forest, ada boosting regressor,

(RBF model, polynomial model, and linear model), and Linear Regression. The Prediction of crude oil rates based on the previous datasets on the data and prices as the feature list are inputs and target list are predicted values.

The implementation was on the logistic regression model

which is feasible to some extend for the prediction of the

petrol prices. The implementation is on predicting the

crude oil prices for the days using linear regression Python

machine learning Algorithm and plotting the graph based on

prediction

We have chosen Linear Regression which is the best fit

among Random Forest, Ada boost regressor (rbf

model, polynomial model, and linear model), and Linear

Regression. The predictions are most approximate with

Linear Regression Algorithm. The algorithm automatically

uses the kernel function that is most appropriate to the data.

SVR uses the linear kernel when there are many attributes

(approx. 100) in the training data, otherwise it uses the

Gaussian Kernel. In the proposed system we have taken the

datasets which has the Crude oil price and diesel price.

Based on the dataset we have made feature list and target list

where the target list is price value of diesel and feature list is

the Petrol price. After the analysis of data is done we

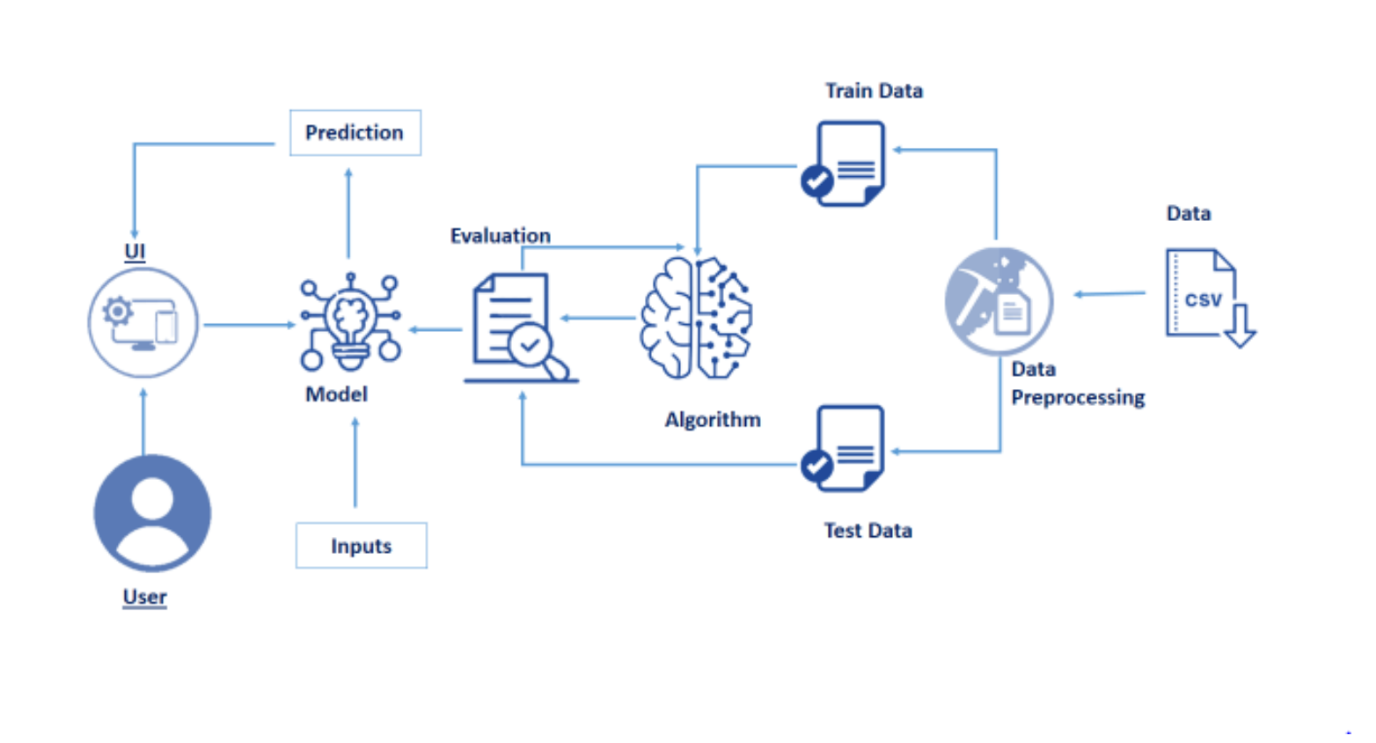
have fitted both feature list and target list using python

Machine Learning Linear Algorithm and predicted the

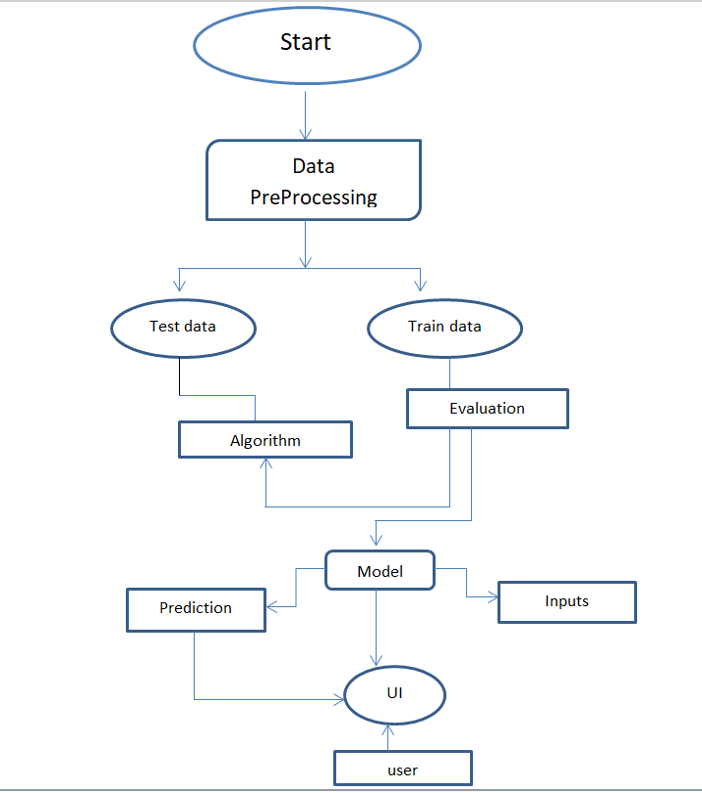
values for feature list from the dataset values.

After making an analysis of what data to be collected according to the prediction of oil price we have finalised the first attribute to be crude oil price in the place from where we import the crude oil (Dubai). Where the transportation also makes a difference in oil price we have also taken the crude oil price of India relating to crude oil price of Dubai. So, we have finalised the attributes which are crude oil price in Dubai and crude oil price of India. we have also taken diesel price of Dubai into consideration which may help in predicting the diesel price of India accurately. After the attribute selection data cleaning is done according to the dataset which we have collected like removing the Null values. The mandatory thing which we need to after collecting the attributes is to know the correlation among the attribute.

### Block Diagram

[](https://github.com/smartinternz02/SI-GuidedProject-4884-1627462177/blob/main/Images/Technical%20Architecture.png)

## Flow Chart

[](https://github.com/smartinternz02/SI-GuidedProject-4884-1627462177/blob/main/Images/Flow%20chart.png)

BACKGROUND

Regression analysis is a machine learning approach that aims to accurately predict the value of continuous output variables from certain independent input variables, via automatic estimation of their latent relationship from data.

**Algorithms**

The algorithms used in this project are as following:

**Linear Regression:**

Multiple linear regression model will be expressed as followed:

y = a0 + a1x + a2x2+....+e

y is the dependent variable and x is the independent variable, a0 is the constant term, is the intercept of the regression line on the vertical axis and a1 is the regression coefficient that is the slope of the regression line. e is the random error which will be used to express the effect of random factors on dependent variable [8]. Step wise algorithm is as follows:

STEP 1: IMPORTING LIBRARIES AND LOADING THE DATA.

Import the libraries that might be required to build our model. To get started we imported pandas, Matplotlib, numpy etc. After importing the libraries, next step will be fetching the dataset and loading our data. The format of the data should be (.csv/.xls).

STEP 2: VISUALISING THE DATA

Visualising the data is important in order to find any correlation between the different parameters. Matplotlib is excellent library that can be used to visualize our data on various different plots.

STEP 3: FEATURE ENGINEERING

When we visualize our data, we found that there is a strong correlation between the two parameters: date and price. Thereby we will be using these parameters for building our model.

STEP 4: FITTING THE LINEAR REGRESSION MODEL

After that import the method train\_test\_split from sklearn library. This is used to split our data into training and testing data. Commonly 70–80% of the data is taken as the training dataset while the remaining data constitutes the testing dataset. After that the intercept and coefficient of our model can be calculated.

**Decision Tree Algorithm(For Regression):**

A decision tree represents a tree-structured classifier that performs a split test in its internal node and predicts a target class of an example in its leaf node.[9]. Decision trees build regression or classification models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes. Consider Fig 1 where X1 and X2 as independent variables and Y as dependent variables. Fig 2 represents the decision tree for the scatter plot. Based on the decision tree, the model made the splits. Now if the new data point lies in between X1=50 and X2 it comes under split 4.

The stepwise algorithm is as follows:

STEP 1: IMPORTING LIBRARIES AND LOADING THE DATA.

Import the libraries that might be required to build our model. To get started we imported pandas, Matplotlib, numpy etc. After importing the libraries, next step will be fetching the dataset and loading our data. The format of the data should be (.csv/.xls).

STEP 2: VISUALISING THE DATA

Visualising the data is important in order to find any correlation between the different parameters. Matplotlib is excellent library that can be used to visualize our data on various different plots.

STEP 3: Splitting the dataset into the Training set and Test set

In this step, we have to split the dataset into training set ad test set. We used only 20% of dataset to test the data and remaining 80% of data set used as training set.

STEP 4: Training the Decision Tree Regression model on the training set.

We import the DecisionTreeRegressor class from sklearn.tree and named it as regressor . Then we fit the X\_train and the y\_train to the model by using the regressor.fit function.

Step 5: Predicting the Results

we predict the results of the test set with the model trained on the training set values using the regressor.predict function and assign it to y\_pred. Step 6: Comparing the Real Values with Predicted Values. In this step, we compare and display the values of y\_test as ‘Real Values’ and y\_pred as ‘Predicted Values’.

**Random Forest Algorithm(For Regression):**

Random forests are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest[10]. Random forest uses ensemble learning. It uses decision trees for n times and predicts the output.

Step 1: IDENTIFIES THE DEPENDENT (Y) AND INDEPENDENT VARIABLES (X)

Dependent variable will be prices while independent variables are the remaining columns left in the dataset.

Step 2: SPLIT THE DATASET INTO THE TRAINING SET AND TEST SET The training and test split are very important. The training set contains known output from which the model learns off of. The test set then tests the model’s predictions based on what it learned from the training set.

Step 3: TRAINING THE RANDOM FOREST REGRESSION MODEL ON THE WHOLE DATASET

From the sklearn package we import the class RandomForestRegressor, create an instance of it, and assign it to a variable. The parameter n\_estimators creates n number of trees in your random forest. The .fit() function allows us to train the model, adjusting weights according to the data values in order to achieve better accuracy. After training, then our model is ready to make predictions, which is called by the .predict() method.

Step 4: PREDICTING THE TEST SET RESULTS

Now our random forest model is successfully created. R² score tells us how well our model is fitted to the data by comparing it to the average line of the dependent variable. If the score is nearer to 1, then it means that our model performs well, if the score is farther from 1, then it means that our model does not perform well.

Implementation Results

The whole project is based on python, machine learning, and flask.

**Data Collection:**

We collected the data regarding the petrol prices among countries from Kaggle.

**Data Pre-processing:**

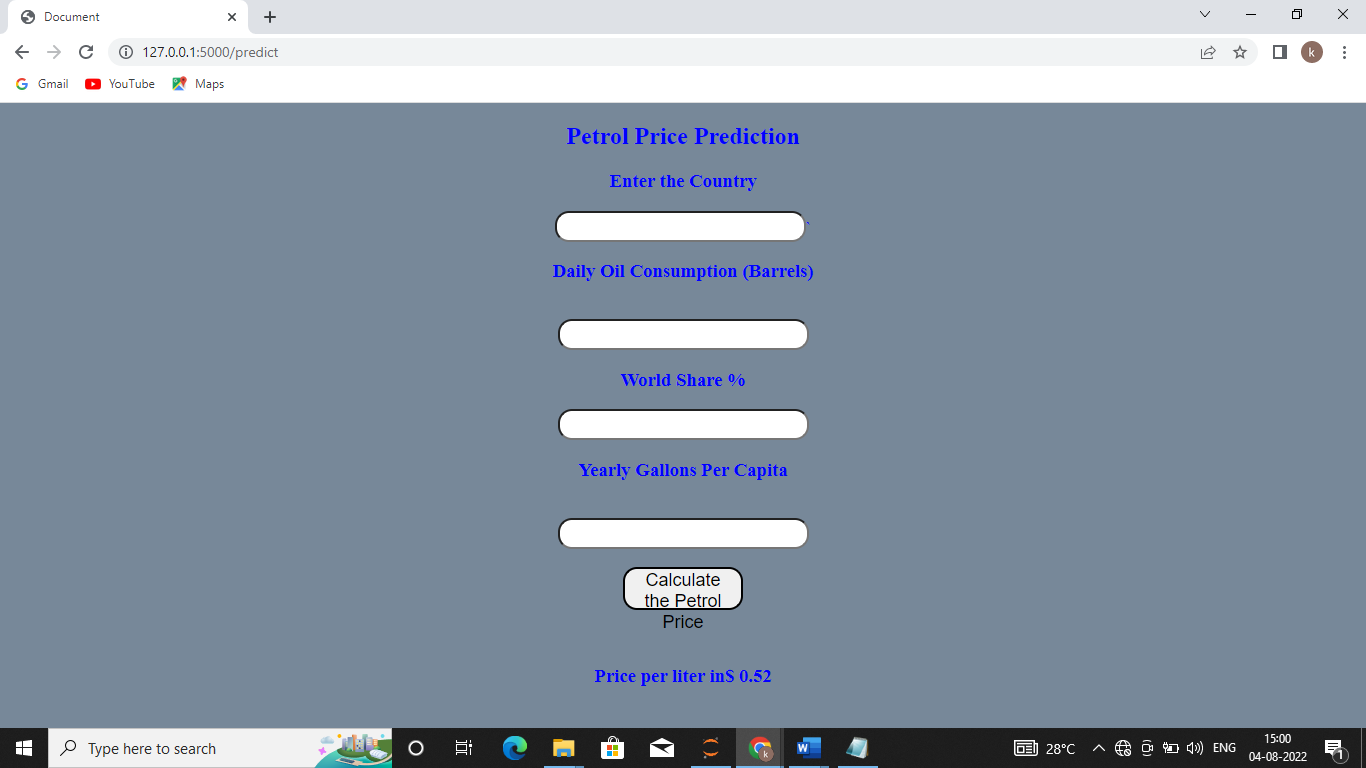
The raw data cannot be used directly for training the model. Hence we perform pre-processing on the raw data. First we import the libraries which are frequently used in our project. Here we have used numpy which mainly focuses on operations on arrays, matplotlib for plots and pandas to work on data. After importing the libraries we import our data set. There may be a chance of missing data in the dataset. Missing data may deviate our results. In order to avoid this we use the SimpleInputer class to replace missing data with mean. Based on the data set and dependent variables we replace missing values will mean, median, constant number etc. If the data set is too heavy and there is only 1% of missing data we ignore the rows with missing values. Now we encode the categorical data using OneHotEncoder class. This method transforms the categorical variable into a set of binary variables (also known as dummy variables). It used N-1 features to show N labels. This improves the machine to understand the data. Next we split the data into a testing set and training set. We apply machine learning algorithms on the training set. In our project we used random forest regression, decision tree regression, simple linear regression. We give the train set as input for the models and train them.

**User Interface and output:**

A user interface improves the usage of the model. We use flask and HTML to build the user interface. From the above metrics results, we conclude that Random Forest Regression has less error. Hence we use Random Forest Regression in our project. The predicted value will be in Dollars.



**Figure 1: User Interface of giving Input**

****

**Figure 2: User Interface of getting output**

# Performance Metrics

# In order to evaluate the best model, we useperformance metrics namely, root mean square error, mean absolute error and R2 score. Based on these error metric values, we pick the best model. Those error metrics for regression are as follows :

**Root Mean Square Error (RMSE):**

The RMSE can be calculated as follows:

RMSE = sqrt(1 / N \* (sum for i to N (exp\_i – pred\_i)2 )

Where exp\_i is the i’th expected value in the dataset, pred\_i is the i’th predicted value, and sqrt() is the square root function We can say that:

RMSE = sqrt(MSE)

**Mean Absolute Error (MAE) :**

The average of the absolute error values is called mean absolute error.

MAE = 1 / N \* (sum for i to N abs(exp\_i – pred\_i))

Where exp\_i is the i’th expected value in the dataset, pred\_i is the i’th predicted value and abs() is the absolute function. A perfect mean absolute error value is 0.0, which suggests that all predictions match with the expected values exactly.

**R2 Score - Coefficient of Determination:**

It is the quantity of the variation within the output dependent attribute which is predictable from the input independent variable(s). The best possible score is 1 which is obtained when the anticipated values are an equivalent because of the actual values.

CONCLUSION

Machine learning is one of the techniques of Artificial Intelligence which is used for extracting valuable knowledge from large databases. Among all the models used, ADA boost Regressor gave us the best results. The error is very less when compared to other regression models. The proposed system can accurately predict the prices of crude oil which helps us to buy crude oil in advance and decrease the expenses spent.

REFERENCES

* <https://labs.openviewpartners.com/>
* Source data related to our analysis has been collected from
* https://[www.kaggle.com/](http://www.kaggle.com/) and other you tube channels

CODE SAMPLE

import pandas as pd  
import numpy as np  
from sklearn.model\_selection import train\_test\_split  
from sklearn.tree import DecisionTreeRegressor  
from sklearn.ensemble import RandomForestRegressor  
from sklearn.ensemble import GradientBoostingRegressor  
from sklearn.ensemble import AdaBoostRegressor  
from flask import Flask, render\_template, request

df=pd.read\_csv('Petrol Dataset June 20 2022.csv',encoding='latin-1')

df.head()

# Country Daily Oil Consumption (Barrels) World Share \  
0 1 United States 19687287 20.30%   
1 2 China 12791553 13.20%   
2 3 India 4443000 4.60%   
3 4 Japan 4012877 4.10%   
4 5 Russia 3631287 3.70%   
  
 Yearly Gallons Per Capita Price Per Gallon (USD) Price Per Liter (USD) \  
0 934.3 5.19 1.37   
1 138.7 5.42 1.43   
2 51.4 5.05 1.33   
3 481.5 4.69 1.24   
4 383.2 3.41 0.90   
  
 Price Per Liter (PKR)   
0 289.97   
1 302.87   
2 281.93   
3 262.05   
4 190.56

df.info()

<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 181 entries, 0 to 180  
Data columns (total 8 columns):  
 # Column Non-Null Count Dtype   
--- ------ -------------- -----   
 0 # 181 non-null int64   
 1 Country 181 non-null object   
 2 Daily Oil Consumption (Barrels) 181 non-null int64   
 3 World Share 181 non-null object   
 4 Yearly Gallons Per Capita 181 non-null float64  
 5 Price Per Gallon (USD) 181 non-null float64  
 6 Price Per Liter (USD) 181 non-null float64  
 7 Price Per Liter (PKR) 181 non-null float64  
dtypes: float64(4), int64(2), object(2)  
memory usage: 11.4+ KB

df.shape

(181, 8)

df.isnull().sum()

# 0  
Country 0  
Daily Oil Consumption (Barrels) 0  
World Share 0  
Yearly Gallons Per Capita 0  
Price Per Gallon (USD) 0  
Price Per Liter (USD) 0  
Price Per Liter (PKR) 0  
dtype: int64

#drop column  
df.drop('#',inplace =True , axis=1)

#drop column  
df.drop('Price Per Liter (PKR)',inplace =True , axis=1)

df.head()

Country Daily Oil Consumption (Barrels) World Share \  
0 United States 19687287 20.30%   
1 China 12791553 13.20%   
2 India 4443000 4.60%   
3 Japan 4012877 4.10%   
4 Russia 3631287 3.70%   
  
 Yearly Gallons Per Capita Price Per Gallon (USD) Price Per Liter (USD)   
0 934.3 5.19 1.37   
1 138.7 5.42 1.43   
2 51.4 5.05 1.33   
3 481.5 4.69 1.24   
4 383.2 3.41 0.90

Rename column **World Share** to **World Share %**

#datatype of this column is str,so we need to change it to float, and remove the % symbol. The % symbol in title will give the information the this column contains percentages.  
df.rename(columns={'World Share': 'World Share %'},inplace=True)

df.head()

Country Daily Oil Consumption (Barrels) World Share % \  
0 United States 19687287 20.30%   
1 China 12791553 13.20%   
2 India 4443000 4.60%   
3 Japan 4012877 4.10%   
4 Russia 3631287 3.70%   
  
 Yearly Gallons Per Capita Price Per Gallon (USD) Price Per Liter (USD)   
0 934.3 5.19 1.37   
1 138.7 5.42 1.43   
2 51.4 5.05 1.33   
3 481.5 4.69 1.24   
4 383.2 3.41 0.90

Convert the data type of **World share** column to **float64**

#The World share % is not exactly accurate therefor we recalculate the World share.  
#Each value from 'Daily Oil Consumption (Barrels)' divided by summation of Daily Oil Consumption (Barrels) multiply by 100.  
Total\_Consumption=df['Daily Oil Consumption (Barrels)'].sum()  
df['World Share %']=(df['Daily Oil Consumption (Barrels)']/Total\_Consumption)\*100

df.head()

Country Daily Oil Consumption (Barrels) World Share % \  
0 United States 19687287 20.385127   
1 China 12791553 13.244965   
2 India 4443000 4.600487   
3 Japan 4012877 4.155118   
4 Russia 3631287 3.760002   
  
 Yearly Gallons Per Capita Price Per Gallon (USD) Price Per Liter (USD)   
0 934.3 5.19 1.37   
1 138.7 5.42 1.43   
2 51.4 5.05 1.33   
3 481.5 4.69 1.24   
4 383.2 3.41 0.90

Linear Regression Model

df.columns

Index(['Country', 'Daily Oil Consumption (Barrels)', 'World Share %',  
 'Yearly Gallons Per Capita', 'Price Per Gallon (USD)',  
 'Price Per Liter (USD)'],  
 dtype='object')

dum=pd.get\_dummies(df.Country)

dum['Daily Oil Consumption (Barrels)']=df['Daily Oil Consumption (Barrels)']

dum['World Share %']=df.iloc[:,2:3]

dum['Yearly Gallons Per Capita']=df['Yearly Gallons Per Capita']

dum['Price Per Gallon (USD)']=df['Price Per Gallon (USD)']

dum.head()

Afghanistan Albania Algeria Angola Argentina Armenia Aruba \  
0 0 0 0 0 0 0 0   
1 0 0 0 0 0 0 0   
2 0 0 0 0 0 0 0   
3 0 0 0 0 0 0 0   
4 0 0 0 0 0 0 0   
  
 Australia Austria Azerbaijan ... Vanuatu Venezuela Vietnam Yemen \  
0 0 0 0 ... 0 0 0 0   
1 0 0 0 ... 0 0 0 0   
2 0 0 0 ... 0 0 0 0   
3 0 0 0 ... 0 0 0 0   
4 0 0 0 ... 0 0 0 0   
  
 Zambia Zimbabwe Daily Oil Consumption (Barrels) World Share % \  
0 0 0 19687287 20.385127   
1 0 0 12791553 13.244965   
2 0 0 4443000 4.600487   
3 0 0 4012877 4.155118   
4 0 0 3631287 3.760002   
  
 Yearly Gallons Per Capita Price Per Gallon (USD)   
0 934.3 5.19   
1 138.7 5.42   
2 51.4 5.05   
3 481.5 4.69   
4 383.2 3.41   
  
[5 rows x 185 columns]

x=dum

y=df['Price Per Liter (USD)']

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

from sklearn.linear\_model import LinearRegression

model=LinearRegression()

model.fit(train\_x,train\_y)

LinearRegression()

prediction=model.predict(test\_x)

prediction

array([1.37981013, 1.68026906, 1.29462075, 0.70592437, 1.39263966,  
 2.30504004, 0.51585672, 1.25500392, 1.36863827, 1.50587104,  
 0.9942878 , 1.35269455, 1.35799953, 1.78323234, 0.47621335,  
 1.50604831, 2.00790347, 1.14407923, 1.68049539, 2.18504243,  
 2.018471 , 1.6805655 , 1.04112592, 1.03048313, 2.02107365,  
 1.26323647, 0.03285635, 2.20604603, 1.21569722, 0.95205666,  
 1.03844995, 0.42884898, 1.019949 , 0.57630612, 0.57744787,  
 2.20928578, 1.33170989])

GradientBoostingRegressor

est = GradientBoostingRegressor(n\_estimators=100, learning\_rate=0.1,max\_depth=1, random\_state=0,)  
est.fit(train\_x,train\_y)

GradientBoostingRegressor(max\_depth=1, random\_state=0)

est.predict(test\_x)

array([1.36595757, 1.64518631, 1.27590641, 0.72032364, 1.36595757,  
 2.25143367, 0.55622712, 1.26554099, 1.36595757, 1.52732507,  
 0.97297402, 1.3594668 , 1.3594668 , 1.77160896, 0.49676711,  
 1.52732507, 2.00606267, 1.1612475 , 1.64518631, 2.16800375,  
 2.00606267, 1.64518631, 1.05211525, 1.05211525, 2.00606267,  
 1.26554099, 0.18483408, 2.18729229, 1.23655807, 0.96334919,  
 1.05211525, 0.45906577, 1.05211525, 0.61155736, 0.61155736,  
 2.18729229, 1.3594668 ])

from sklearn.ensemble import GradientBoostingRegressor,AdaBoostRegressor,RandomForestRegressor

ada = AdaBoostRegressor(random\_state=0, n\_estimators=100).fit(train\_x,train\_y)

ada.predict(test\_x)

array([1.43777778, 1.68183673, 1.30826087, 0.52277778, 1.43777778,  
 2.12217391, 0.52277778, 1.30826087, 1.43777778, 1.43777778,  
 0.91454545, 1.43777778, 1.43777778, 1.68183673, 0.52277778,  
 1.43777778, 2.12217391, 1.20465116, 1.68183673, 2.12217391,  
 2.12217391, 1.68183673, 1.20465116, 0.91454545, 2.12217391,  
 1.30826087, 0.27222222, 2.12217391, 1.20465116, 0.91454545,  
 1.20465116, 0.52277778, 0.91454545, 0.52277778, 0.52277778,  
 2.12217391, 1.43777778])

ada.score(train\_x,train\_y)

0.9899631217179619

ada.score(test\_x,test\_y)

0.9694884292245116

import pickle

file = open('petrol.pkl','wb')  
pickle.dump(ada,file)

file.close()

file = open('petrol.pkl','rb')

# loading the binary file

modelf = pickle.load(file)

modelf.score(test\_x,test\_y)

0.9694884292245116

dum.columns

Index(['Afghanistan', 'Albania', 'Algeria', 'Angola', 'Argentina', 'Armenia',  
 'Aruba', 'Australia', 'Austria', 'Azerbaijan',  
 ...  
 'Vanuatu', 'Venezuela', 'Vietnam', 'Yemen', 'Zambia', 'Zimbabwe',  
 'Daily Oil Consumption (Barrels)', 'World Share %',  
 'Yearly Gallons Per Capita', 'Price Per Gallon (USD)'],  
 dtype='object', length=185)

df5 = pd.DataFrame(df.iloc[0],columns=df.iloc[0].index,index=[0])  
filename = 'petrol.pkl'  
modelf = pickle.load(open(filename, 'rb'))  
app = Flask(\_\_name\_\_)  
df=pd.read\_csv('Petrol Dataset June 20 2022.csv',encoding='latin-1')  
@app.route('/',methods=['GET'])  
def index():  
 return render\_template('index.html')  
@app.route('/predict', methods=['POST'])  
def predict():  
 temp\_array = list()  
 country = (request.form['Country'])  
  
 rat = float(request.form['Daily Oil Consumption (Barrels)'])  
 rev = float(request.form['World Share %'])  
 pri = float(request.form['Yearly Gallons Per Capita'])  
 gal=2.19  
 temp\_array = temp\_array + [rat,rev,pri,gal]  
 data = np.array([temp\_array])  
 pred = float(modelf.predict(data)[0])  
 #pred = modelf.predict([[country,rat,rev,pri,gal]])  
 output=round(pred,2)  
 return render\_template('index.html',ds = df5.to\_html(),prediction\_text="Price per liter in$ {}".format(output))  
app.run()

\* Serving Flask app "\_\_main\_\_" (lazy loading)  
 \* Environment: production  
 WARNING: This is a development server. Do not use it in a production deployment.  
 Use a production WSGI server instead.  
 \* Debug mode: off

\* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)  
127.0.0.1 - - [04/Aug/2022 14:53:14] "GET / HTTP/1.1" 200 -  
127.0.0.1 - - [04/Aug/2022 14:55:24] "GET / HTTP/1.1" 200 -  
127.0.0.1 - - [04/Aug/2022 15:00:32] "POST /predict HTTP/1.1" 200 -