# NATURAL GAS PRICE PREDICTION

## 1. INTRODUCTION

### 1.1. Overview

Forecasting natural gas prices is a powerful and essential tool which has become more important for different stakeholders in the natural gas market, allowing them to make better decisions for managing the potential risk, reducing the gap between the demand and supply, and optimizing the usage of resources based on accurate predictions.

### 1.2. Purpose

Accurate natural gas price forecasting not only provides an important guide for effective implementation of energy policy and planning, but also is extremely significant in economic planning, energy investment, and environmental conservation.
The aim of this project is to build data-driven machine learning models for natural gas price forecasting.

## 2. LITERATURE SURVEY

### 2.1. Existing Problem

As of mid-2020, natural gas production in the US had peaked three times, with current levels exceeding both previous peaks. It reached 24.1 trillion cubic feet per year in 1973, followed by a decline, and reached 24.5 trillion cubic feet in 2001. After a brief drop, withdrawals increased nearly every year since 2006 (owing to the shale gas boom), with 2017 production at 33.4 trillion cubic feet and 2019 production at 40.7 trillion cubic feet. After the third peak in December 2019, extraction continued to fall from March onward due to decreased demand caused by the COVID-19 pandemic in the US. The 2021 gloabal energy crisis was driven by a global surge in demand as the world quit the economic recession caused by COVID-19, particularly due to strong energy demand in Asia.

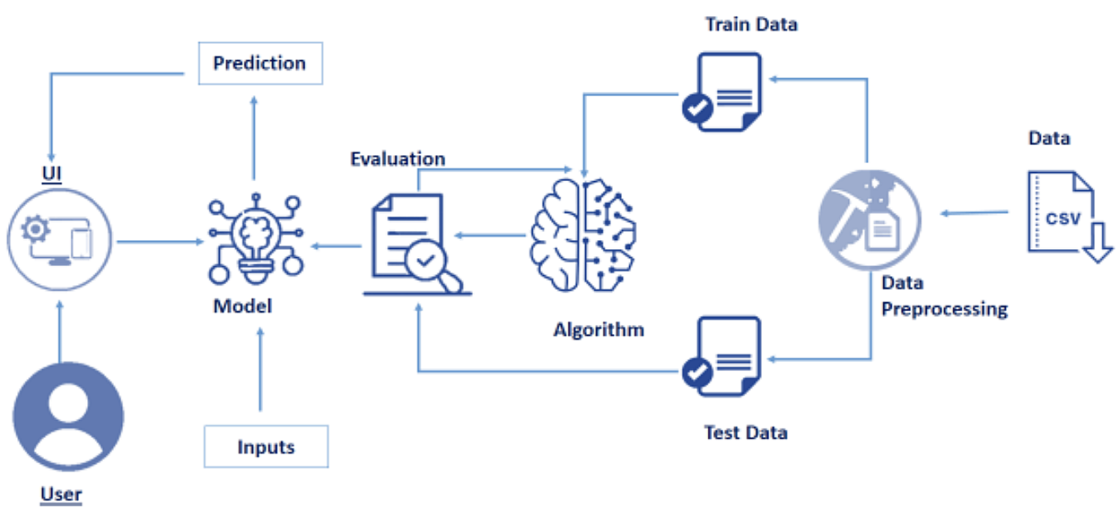
The unfortunate history of energy cost determining models has urged experts to go to different wellsprings of data about future energy costs, including most unmistakably, energy prospects markets. Energy prospects markets are 'center points' that cost and market flammable gas. Dividers (1995), looking at quite a while of spot costs views that as, by and large, gas fates are fair-minded indicators of future spot costs though Herbet (1993) finds predisposition in petroleum gas prospects costs where fates costs are more noteworthy than acknowledged spot costs. Chinn et al (2005) views fates costs as fair indicators of future spot costs, except for those in the gaseous petrol market at the multi month skyline and they marginally beat time series models.

**2.2. Proposed Solution**

Utilizing the informational collection of costs from 7th of January 1997 until 11th of august 2020, we will be attempting to foresee the costs of petroleum gas by testing through different AI models and giving a continuous electronic GUI to request that the client enter the ideal date to anticipate the pace of the flammable gas on that specific day. This study expands upon the current writing by examining the precision of different estimate techniques until the best fit skyline is reached.

## 3. THEORETICAL ANALYSIS

### 3.1. Block Diagram




### 3.2. Software Designing

We will be using the following modules and softwares to help build the project:

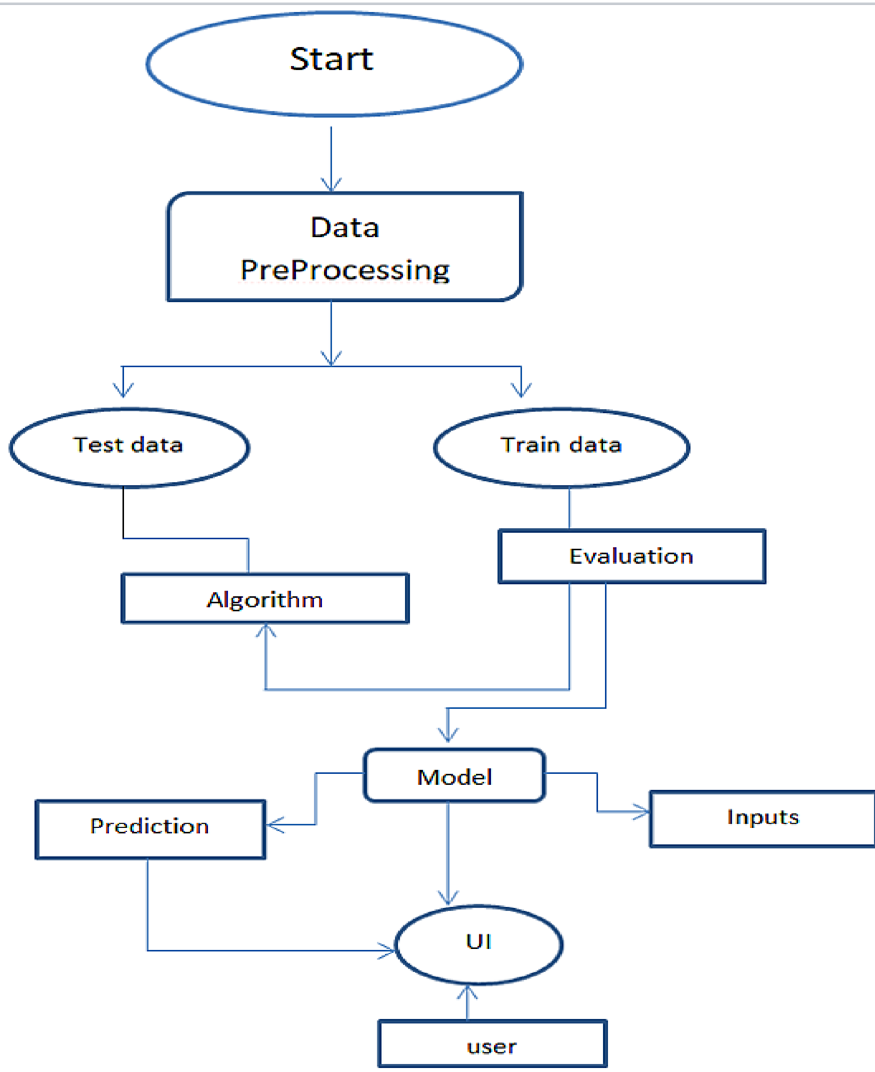
- pandas 1.2.4
- matplotlib 3.4.2
- pydotplus 2.0.2
- six 1.15.0
- ipython 7.24.1
- scikit-learn 0.24.2
- seaborn~=0.11.1

- scipy~=1.6.3
- Jupyter Notebook
- Spyder
- IBM Watson Cloud along with Machine Learning module.

## 4. EXPERIMENTAL INVESTIGATIONS

The assessment of the proposed approach is Natural gas cost forecast by considering day, month, year on the dataset. In this the undertaking preprocessing and preparing of the informational collection is finished by utilizing a jupyter notebook. After model structure we fabricate a web application utilizing flask system.

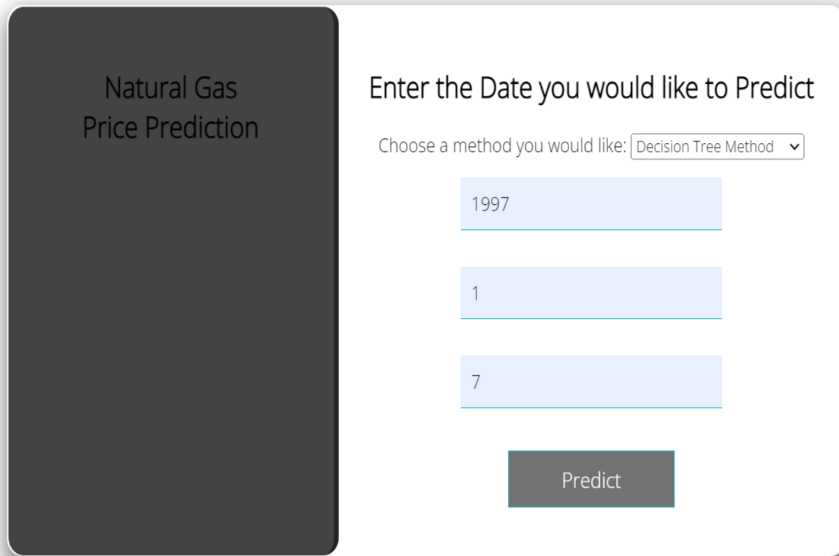
## 5. FLOWCHART

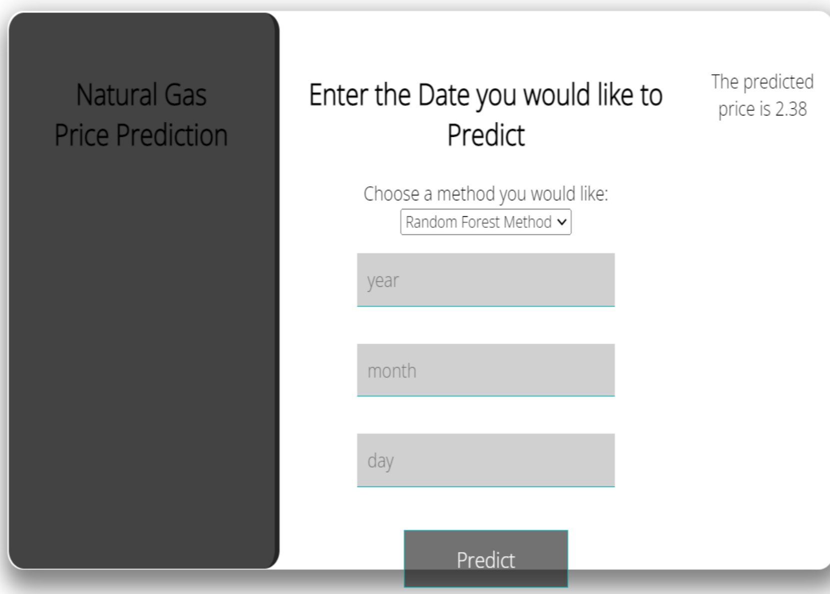


# NATURAL GAS PRICE PREDICTION

## 6. RESULT

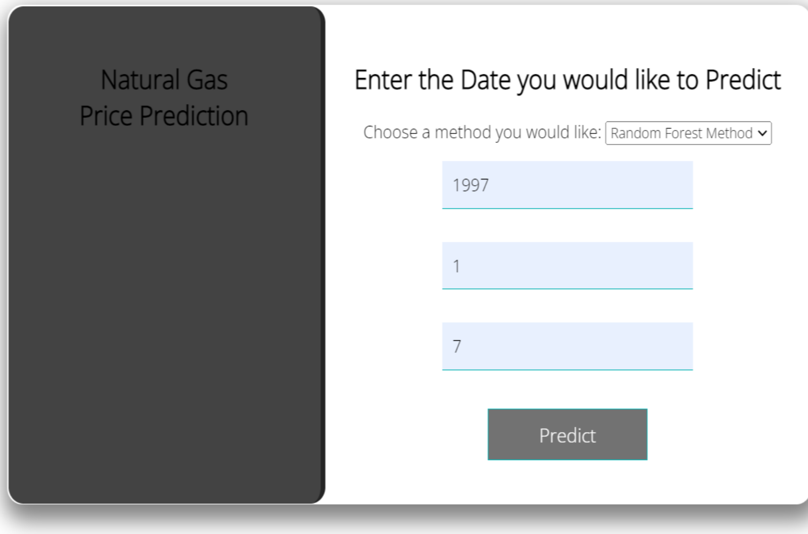
### 6.1. Decision Tree

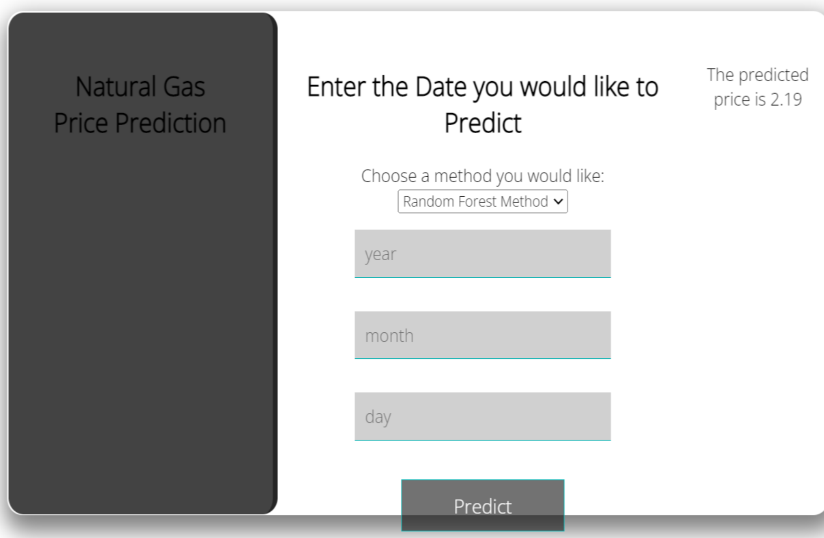


# NATURAL GAS PRICE PREDICTION

## 6.2. Random Forest



# 7. ADVANTAGES AND DISADVANTAGES

## 7.1. Decision Tree

### 7.1.1. Advantages

- The decision tree model has strong generalization ability, can be trained fast, and is not sensitive to missing data.
- The decision tree model can be used for both classification and regression problems, and it is easy to interpret, understand and visualize.
- Compared with other algorithms, data preparation during pre-processing in a decision tree requires less effort and does not require normalization of data.
- A decision tree is one of the quickest ways to identify relationships between variables and the most significant variable.
- The output of a decision tree can also be easily understood.
- A Decision tree model is very intuitive and easy to explain to technical teams as well as stakeholders.

### 7.1.2. Disadvantages

- In a decision tree model, if there is noise in the data it is prone to overfitting.
- The numerical calculations involved in a decision tree generally consumes more memory.
- A small change in the data can cause a large change in the structure of the decision tree causing instability.
- For a Decision tree sometimes calculation can go far more complex compared to other algorithms.
- The reproducibility of a decision tree is exceptionally sensitive as even a slight change in data can bring about an enormous change in the tree structure.
- The multifaceted space and time complexity in a decision tree model are generally higher than other decision-making tools.
- Due to this multifaceted complexity, the training time involved in Decision tree model is higher as well. This increases the cost involved in training as well.

## 7.2. Random Forest

### 7.2.1. Advantages

- Random Forest algorithm outputs the importance of features which is a very useful.
- Random Forest algorithm is less prone to overfitting than Decision Tree and other algorithms.
- It enhances the accuracy of the model and prevents the overfitting issue.
- Random Forest works well with both categorical and continuous variables.
- It is capable of handling large datasets with high dimensionality.

### 7.2.2. Disadvantages

- Random Forest is more complex. It’s hard to visualize the model or understand why it predicted something.
- It is a difficult tradeoff between the training time (and space) and increased number of trees. The increase of the number of trees can improve the accuracy of prediction. However, random forest often involves higher time and space to train the model as a larger number of trees are involved.
- Random forest may not get good results for small data or low-dimensional data (data with few features). Since the randomness becomes greatly reduced. Processing high-dimensional data and feature-missing data are the strengths of random forest.
- Random forest is like a black box that we have little control over. Its computations may go far more complex compared to other algorithms.
- It is not easily interpretable-it provides feature importance but it does not provide complete visibility into the coefficients as linear regression.

## 8. APPLICATIONS

Natural gas (likewise called fossil gas or essentially gas) is a normally happening combination of vaporous hydrocarbons comprising fundamentally of methane notwithstanding different more modest measures of other higher alkanes. Generally low degrees of follow gases like carbon dioxide, nitrogen, hydrogen sulfide, and helium are additionally present. Petroleum gas is dry and unscented, so it odorizes, for example, mercaptan, which scents like sulfur or spoiled eggs, is normally added to flammable gas supplies for wellbeing so that breaks can be promptly recognized.

CNG is a cleaner and furthermore less expensive option in contrast to other car energizes like fuel (petrol). By the finish of 2014, there were north of 20 million petroleum gas vehicles around the world, drove by Iran (3.5 million), China (3.3 million), Pakistan (2.8 million), Argentina (2.5 million), India (1.8 million), and Brazil (1.8 million).The energy proficiency is for the most part equivalent to that of gas motors, however lower contrasted and present day diesel motors. Fuel/petroleum vehicles changed over completely to burn regular gasoline endure due to the low pressure proportion of their motors, bringing about an editing of conveyed power while burning normal gasoline (10-15%). CNG-explicit motors, be that as it may, utilize a higher pressure proportion because of this fuel's higher octane number of 120-130.

Natural gas represents 1/4 of the worldwide interest and around 1/3 of the US energy interest. After oil, Natural gas is the most prevailing kind of energy. In this way, being going to further develop petroleum gas request expectation is very important. The precise expectation of energy cost is basic to the energy market direction, and it can give a reference to policymakers and market members. By and by, energy costs are impacted by outside factors, and their exact expectation is challenging.Being ready to conjecture flammable gas cost benefits different partners and has turned into a truly significant device for all market members in cutthroat petroleum gas markets. AI predictions have continuously become famous devices for natural gas cost guaging.

# NATURAL GAS PRICE PREDICTION

## 9. CONCLUSION

It has forever been a troublesome undertaking to foresee the specific everyday cost of flammable gas cost. Many factors like political occasions, general monetary circumstances, and merchants' assumptions might affect it. In any case, here, in light of the at various times characteristics, we had the option to accomplish up to 97% precision in foreseeing the cost of some random date. Yet, its difficult to foresee unforeseen situations like demonstrations of fighting or illegal intimidation. Be that as it may, the advantages of having dependable data of what the cost of petroleum gas could be at some random time is fundamental, it could represent the deciding moment economies. Also, for this situation, as this undertaking brings up information driven AI models merit all the consideration it might at any point earn and, surprisingly, more.

## 10. FUTURE SCOPE

The undertaking has been fabricated utilizing 2 models of expectation in particular the Decision Tree technique and Random Forest strategy with the exactness score of more than 97% on both the models (97.4% on Decision Tree and 97.74% on Random Forest Method). By doing some further examination and learning the exactness can be elevated upto 100 percent which would be an ideal forecast continuous application which would be substantially more supportive in the exchanging area.

Frequently ignored in past discussions about the eventual fate of energy, petroleum gas is tracking down its place at the core of the energy conversation. Petroleum gas is a significant fuel for numerous end utilizes — power, industry, warming — and is progressively examined as a likely pathway to decreased oil reliance for transportation. Also, the acknowledgment throughout the course of recent years that the producible flighty gas asset in the U.S. is exceptionally huge has strengthened the conversation about flammable gas as a "span" to a low-carbon future.

## 11. BIBLIOGRAPHY


- https://docs.anaconda.com/anaconda/packages/pkg-docs/
- https://towardsdatascience.com/natural-gas-spot-price-prediction-using-artificial-neural-network-56da369b2346
- The Future of Natural Gas | MIT Energy Initiative
- Natural gas - Wikipedia
- Student Dashboard (smartinternz.com)

# NATURAL GAS PRICE PREDICTION

## APPENDIX

## Source Code:

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
```

In [2]:
```
data=pd.read_csv('daily_csv.csv')
```

In [3]:
```
data
```

Out[3]:

| | Date | Price |
|---|---|---|
| **0** | 1997-01-07 | 3.82 |
| **1** | 1997-01-08 | 3.80 |
| **2** | 1997-01-09 | 3.61 |
| **3** | 1997-01-10 | 3.92 |
| **4** | 1997-01-13 | 4.00 |
| **...** | ... | ... |
| **5933** | 2020-08-05 | 2.23 |
| **5934** | 2020-08-06 | 2.26 |
| **5935** | 2020-08-07 | 2.15 |
| **5936** | 2020-08-10 | 2.18 |

In [4]:
```
data.isnull().any()
```

Out[4]:
```
Date      False
Price      True
dtype: bool
```

In [5]:
```
data.head()
```

Out[5]:

| | Date | Price |
|---|---|---|
| **0** | 1997-01-07 | 3.82 |
| **1** | 1997-01-08 | 3.80 |
| **2** | 1997-01-09 | 3.61 |
| **3** | 1997-01-10 | 3.92 |
| **4** | 1997-01-13 | 4.00 |

In [6]:
```
data.tail()
```

Out[6]:

| | Date | Price |
|---|---|---|
| **5933** | 2020-08-05 | 2.23 |
| **5934** | 2020-08-06 | 2.26 |
| **5935** | 2020-08-07 | 2.15 |

In [7]:
```
data['year'] = pd.DatetimeIndex(data['Date']).year
data['month'] = pd.DatetimeIndex(data['Date']).month
data['day'] = pd.DatetimeIndex(data['Date']).day
```

In [8]:
```
data.drop('Date',axis=1,inplace=True)
```

In [9]:
```
data
```

Out[9]:

| | Price | year | month | day |
|---|---|---|---|---|
| **0** | 3.82 | 1997 | 1 | 7 |
| **1** | 3.80 | 1997 | 1 | 8 |
| **2** | 3.61 | 1997 | 1 | 9 |
| **3** | 3.92 | 1997 | 1 | 10 |
| **4** | 4.00 | 1997 | 1 | 13 |
| **...** | ... | ... | ... | ... |
| **5933** | 2.23 | 2020 | 8 | 5 |
| **5934** | 2.26 | 2020 | 8 | 6 |
| **5935** | 2.15 | 2020 | 8 | 7 |
| **5936** | 2.18 | 2020 | 8 | 10 |
| **5937** | 2.19 | 2020 | 8 | 11 |

# NATURAL GAS PRICE PREDICTION

In [10]:
```
data['Price'].fillna(data['Price'].median(),inplace=True)
```

In [11]:
```
data.isnull().any()
```

Out[11]:
```
Price    False
year     False
month    False
day      False
dtype: bool
```

In [12]:
```
data.describe()
```

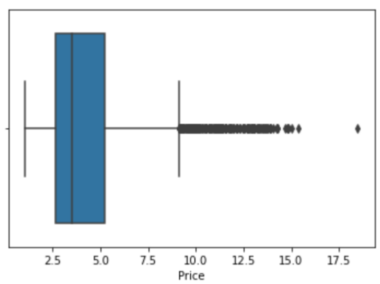
Out[12]:

| | Price | year | month | day |
|---|---|---|---|---|
| **count** | 5938.000000 | 5938.000000 | 5938.000000 | 5938.000000 |
| **mean** | 4.189121 | 2008.366959 | 6.468003 | 15.712193 |
| **std** | 2.191042 | 6.825348 | 3.415981 | 8.742158 |
| **min** | 1.050000 | 1997.000000 | 1.000000 | 1.000000 |
| **25%** | 2.660000 | 2002.000000 | 4.000000 | 8.000000 |
| **50%** | 3.540000 | 2008.000000 | 6.000000 | 16.000000 |
| **75%** | 5.240000 | 2014.000000 | 9.000000 | 23.000000 |
| **max** | 18.480000 | 2020.000000 | 12.000000 | 31.000000 |

In [13]:
```
sns.boxplot(data['Price'])
```

```
C:\Users\HP\anaconda3.x\lib\site-packages\seaborn\_decorators.py:43: FutureWarning: Pass the following variable as a keyword ar
g: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  FutureWarning
```

Out[13]: <AxesSubplot:xlabel='Price'>




In [14]:
```
from scipy import stats
```

In [15]:
```
y=np.abs(stats.zscore(data))
y
```

Out[15]:

| | Price | year | month | day |
|---|---|---|---|---|
| **0** | 0.168482 | 1.665544 | 1.600847 | 0.996656 |
| **1** | 0.177611 | 1.665544 | 1.600847 | 0.882258 |
| **2** | 0.264335 | 1.665544 | 1.600847 | 0.767861 |
| **3** | 0.122838 | 1.665544 | 1.600847 | 0.653463 |
| **4** | 0.086323 | 1.665544 | 1.600847 | 0.310269 |
| **...** | ... | ... | ... | ... |
| **5933** | 0.894226 | 1.704531 | 0.448517 | 1.225452 |
| **5934** | 0.880532 | 1.704531 | 0.448517 | 1.111054 |
| **5935** | 0.930741 | 1.704531 | 0.448517 | 0.996656 |
| **5936** | 0.917048 | 1.704531 | 0.448517 | 0.653463 |
| **5937** | 0.912483 | 1.704531 | 0.448517 | 0.539065 |

5938 rows × 4 columns

# NATURAL GAS PRICE PREDICTION

In [16]:
```
threshold=3
np.where(y>threshold)
```

Out[16]:
```
(array([1534, 1535, 1538, 2164, 2165, 2166, 2167, 2168, 2169, 2170, 2171,
        2174, 2175, 2176, 2177, 2178, 2179, 2180, 2181, 2182, 2183, 2184,
        2185, 2186, 2187, 2188, 2189, 2190, 2191, 2192, 2193, 2194, 2195,
        2196, 2197, 2198, 2199, 2200, 2209, 2210, 2213, 2214, 2215, 2216,
        2217, 2218, 2219, 2220, 2221, 2222, 2223, 2224, 2225, 2226, 2227,
        2228, 2229, 2230, 2231, 2232, 2233, 2234, 2820, 2821, 2822, 2825,
        2826, 2827, 2828, 2829, 2830, 2831, 2832, 2833, 2834, 2835, 2836,
        2837, 2838, 2839, 2840, 2841, 2842, 2843, 2844, 2845, 2846, 2847,
        2848, 2849, 2850, 2851, 2852, 2853, 2854, 2855, 2856, 2857, 2858,
        2859, 2860, 2861, 2862, 2863, 2864, 2865, 2866, 2867, 2868, 2869,
        2870, 2871, 2872, 2873, 2874, 2875, 2876], dtype=int64),
 array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0], dtype=int64))
```

In [17]:
```
df_no_outliers=data[(y<=3).all(axis=1)]
df_no_outliers
```

df_no_outliers

Out[17]:

| | Price | year | month | day |
|---|---|---|---|---|
| **0** | 3.82 | 1997 | 1 | 7 |
| **1** | 3.80 | 1997 | 1 | 8 |
| **2** | 3.61 | 1997 | 1 | 9 |
| **3** | 3.92 | 1997 | 1 | 10 |
| **4** | 4.00 | 1997 | 1 | 13 |
| **...** | ... | ... | ... | ... |
| **5933** | 2.23 | 2020 | 8 | 5 |
| **5934** | 2.26 | 2020 | 8 | 6 |
| **5935** | 2.15 | 2020 | 8 | 7 |
| **5936** | 2.18 | 2020 | 8 | 10 |
| **5937** | 2.19 | 2020 | 8 | 11 |

5821 rows × 4 columns

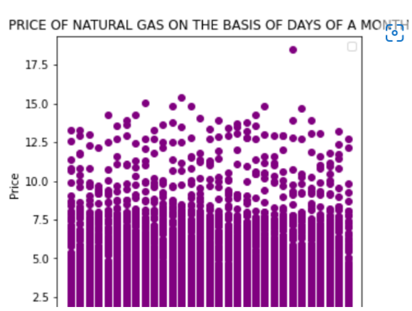
In [18]:
```
df_no_outliers.shape
```

Out[18]:
```
(5821, 4)
```

In [19]:
```
fig=plt.figure(figsize=(5,5))
plt.scatter(data['day'],data['Price'],color='purple')
plt.xlabel('day')
plt.ylabel('Price')
plt.title('PRICE OF NATURAL GAS ON THE BASIS OF DAYS OF A MONTH')
plt.legend()
```

```
No artists with labels found to put in legend.  Note that artists whose label start with an underscore are ignored when legend
() is called with no argument.
```

Out[19]:
```
<matplotlib.legend.Legend at 0x25b517831c8>
```

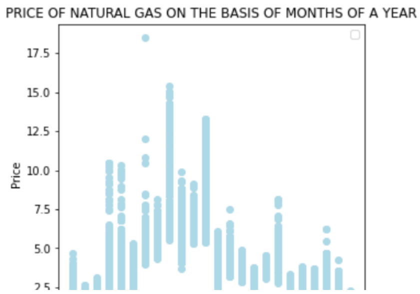


# NATURAL GAS PRICE PREDICTION

In [20]:
```
fig=plt.figure(figsize=(5,5))
plt.scatter(data['year'],data['Price'],color='lightblue')
plt.xlabel('month')
plt.ylabel('Price')
plt.title('PRICE OF NATURAL GAS ON THE BASIS OF MONTHS OF A YEAR')
plt.legend()
```

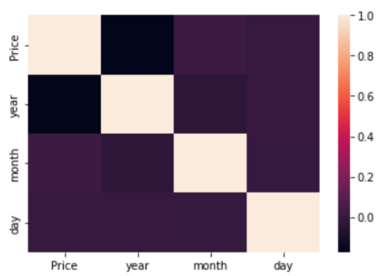
```
No artists with labels found to put in legend.  Note that artists whose label start with an underscore are ignored when legend
() is called with no argument.
```

Out[20]: <matplotlib.legend.Legend at 0x25b51843488>




In [21]:
```
#heat map
sns.heatmap(data.corr())
```

Out[21]: <AxesSubplot:>

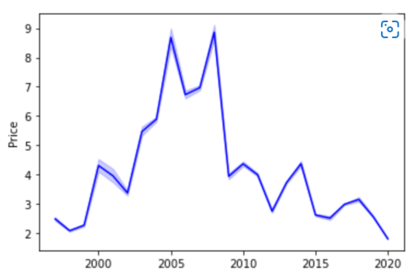



In [22]:
```
plt.bar(data['month'],data['Price'],color='green')
plt.xlabel('month')
plt.ylabel('Price')
plt.title('PRICE OF NATURAL GAS ON THE BASIS OF MONTHS OF A YEAR')
plt.legend()
```

```
No artists with labels found to put in legend.  Note that artists whose label start with an underscore are ignored when legend
```

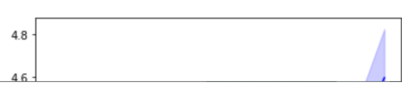
```
sns.lineplot(x='year',y='Price',data=data,color='blue')
```

Out[23]: <AxesSubplot:xlabel='year', ylabel='Price'>




In [24]:
```
sns.lineplot(x='month',y='Price',data=data,color='blue')
```

Out[24]: <AxesSubplot:xlabel='month', ylabel='Price'>



# NATURAL GAS PRICE PREDICTION

## Flask App(app.py)

```python
from flask import Flask, request, render_template
#import pickle
#model = pickle.load(open("model.pkl","rb"))
#model1 = pickle.load(open("model1.pkl","rb"))
import requests
app=Flask(__name__)
# NOTE: you must manually set API_KEY below using information retrieved from your IBM Cloud account.
API_KEY = "C_kKqVayySQGU1SASahc5nnFVmLogVM4SLpj5GsrXx-E"
token_response = requests.post('https://iam.cloud.ibm.com/identity/token', data={"apikey":
 API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
mltoken = token_response.json()["access_token"]

header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}

# NOTE: manually define and pass the array(s) of values to be scored in the next line
#payload_scoring = {"input_data": [{"fields": [array_of_input_fields], "values": [array_of_values_to_be_scored, another_array_o

#response_scoring = requests.post('https://us-south.ml.cloud.ibm.com/ml/v4/deployments/da14b2af-e790-4ef9-aefc-4e03647d8c04/pre
 #headers={'Authorization': 'Bearer ' + mltoken})
#print("Scoring response")
#print(response_scoring.json())
@app.route('/')
def hello():
    return render_template('index.html')
@app.route('/prediction', methods=['GET','POST'])
def prediction():
    a=int(request.form['method'])
    print(a)
    p=int(request.form["year"])
    q=int(request.form["month"])
    r=int(request.form["day"])
    option=[[a,p,q,r]]
    print(option)
```

```python
    r=int(request.form["day"])
    option=[[a,p,q,r]]
    print(option)
    #option = [[int(x) for x in request.form.values()]]
    #p=request.form["year"]
    if option[0][0] == 2:
        del option[0][0]
        x_test = option
        print(x_test)
        payload_scoring = {"input_data": [{"field": [["year", "month", "day"]], "values": x_test}]}
        response_scoring = requests.post('https://us-south.ml.cloud.ibm.com/ml/v4/deployments/121af84c-5fca-429f-b3f3-53fafcba2
        headers={'Authorization': 'Bearer ' + mltoken})
        print("Scoring response")
        pred=response_scoring.json()
        print(pred)
        output=pred['predictions'][0]['values'][0][0]
        print(output)
    else:
        del option[0][0]
        x_test = option
        print(x_test)
        payload_scoring = {"input_data": [{"field": [["year", "month", "day"]], "values": x_test}]}
        response_scoring = requests.post('https://us-south.ml.cloud.ibm.com/ml/v4/deployments/da14b2af-e790-4ef9-aefc-4e03647d8
        headers={'Authorization': 'Bearer ' + mltoken})
        print("Scoring response")
        pred=response_scoring.json()
        print(pred)
        output=pred['predictions'][0]['values'][0][0]
        print(output)
    return render_template('index.html', predic_text='The predicted price is ' + str(output))
if __name__ == "__main__":
    app.run(debug=False)
```