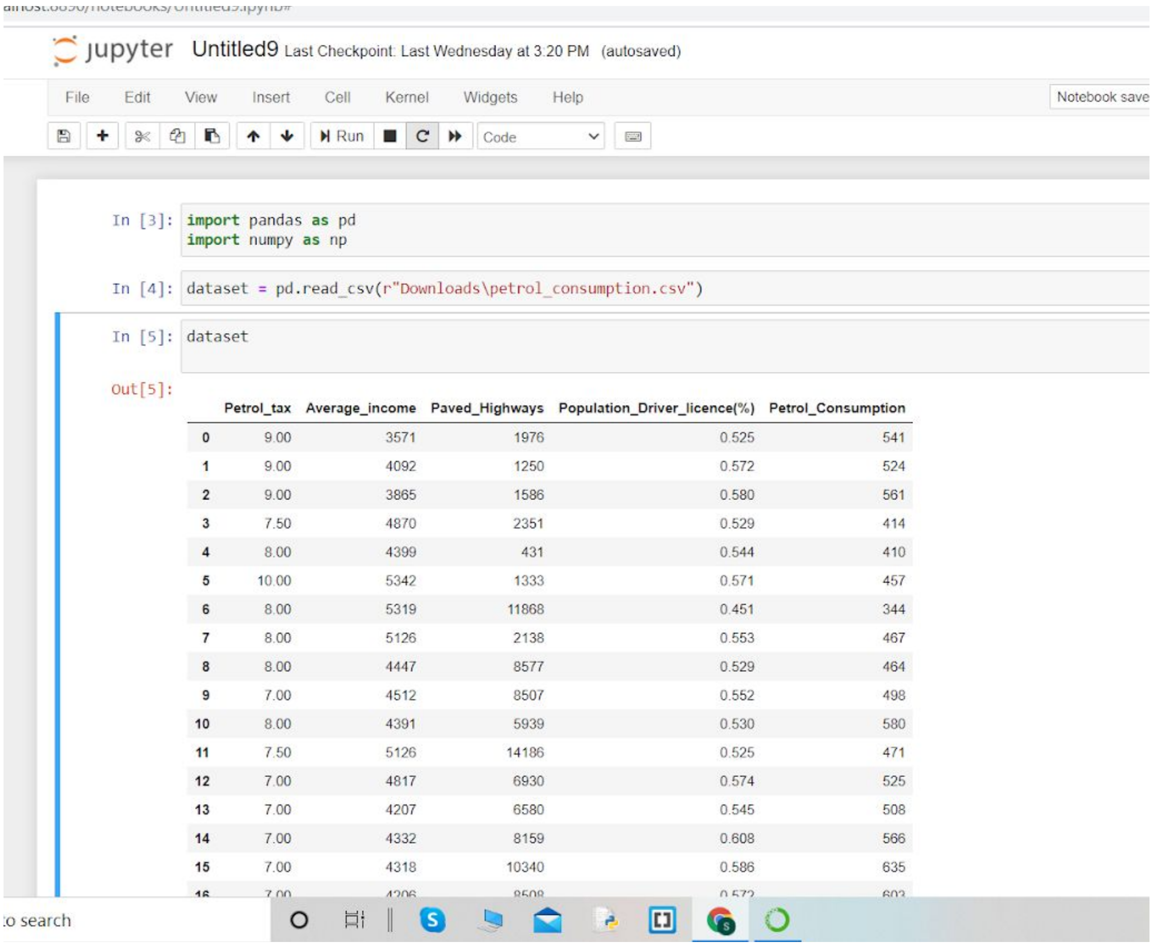

alhost:8890/notebooks/Untitled9.ipynb#
jupyter
Untitled9 Last Checkpoint: Last Wednesday at 3:20 PM (autosaved)
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Notebook save
Run
Code
o search

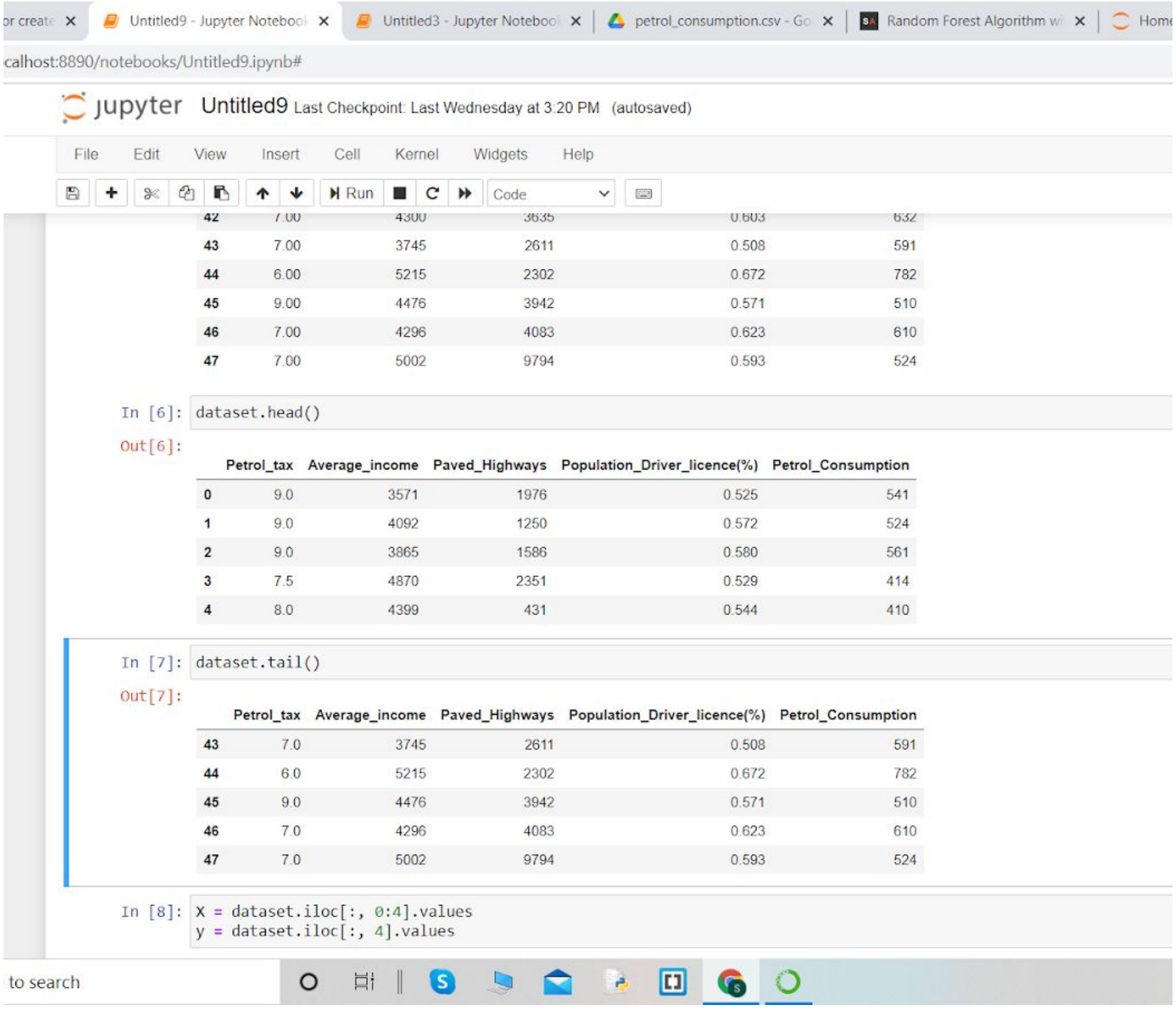
```
In [3]: import pandas as pd
        import numpy as np
```

```
In [4]: dataset = pd.read_csv(r"Downloads\petrol_consumption.csv")
```

```
In [5]: dataset
```

Out[5]:

	Petrol_tax	Average_income	Paved_Highways	Population_Driver_licence(%)	Petrol_Consumption
0	9.00	3571	1976	0.525	541
1	9.00	4092	1250	0.572	524
2	9.00	3865	1586	0.580	561
3	7.50	4870	2351	0.529	414
4	8.00	4399	431	0.544	410
5	10.00	5342	1333	0.571	457
6	8.00	5319	11868	0.451	344
7	8.00	5126	2138	0.553	467
8	8.00	4447	8577	0.529	464
9	7.00	4512	8507	0.552	498
10	8.00	4391	5939	0.530	580
11	7.50	5126	14186	0.525	471
12	7.00	4817	6930	0.574	525
13	7.00	4207	6580	0.545	508
14	7.00	4332	8159	0.608	566
15	7.00	4318	10340	0.586	635
16	7.00	4206	8508	0.572	603



or create
Untitled9 - Jupyter Noteboo
Untitled3 - Jupyter Noteboo
petrol_consumption.csv - Go
Random Forest Algorithm wi
Home
calhost:8890/notebooks/Untitled9.ipynb#
jupyter
Untitled9 Last Checkpoint: Last Wednesday at 3:20 PM (autosaved)
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Run
Code
to search

42	7.00	4300	3635	0.603	632
43	7.00	3745	2611	0.508	591
44	6.00	5215	2302	0.672	782
45	9.00	4476	3942	0.571	510
46	7.00	4296	4083	0.623	610
47	7.00	5002	9794	0.593	524

```
In [6]: dataset.head()
```

Out[6]:

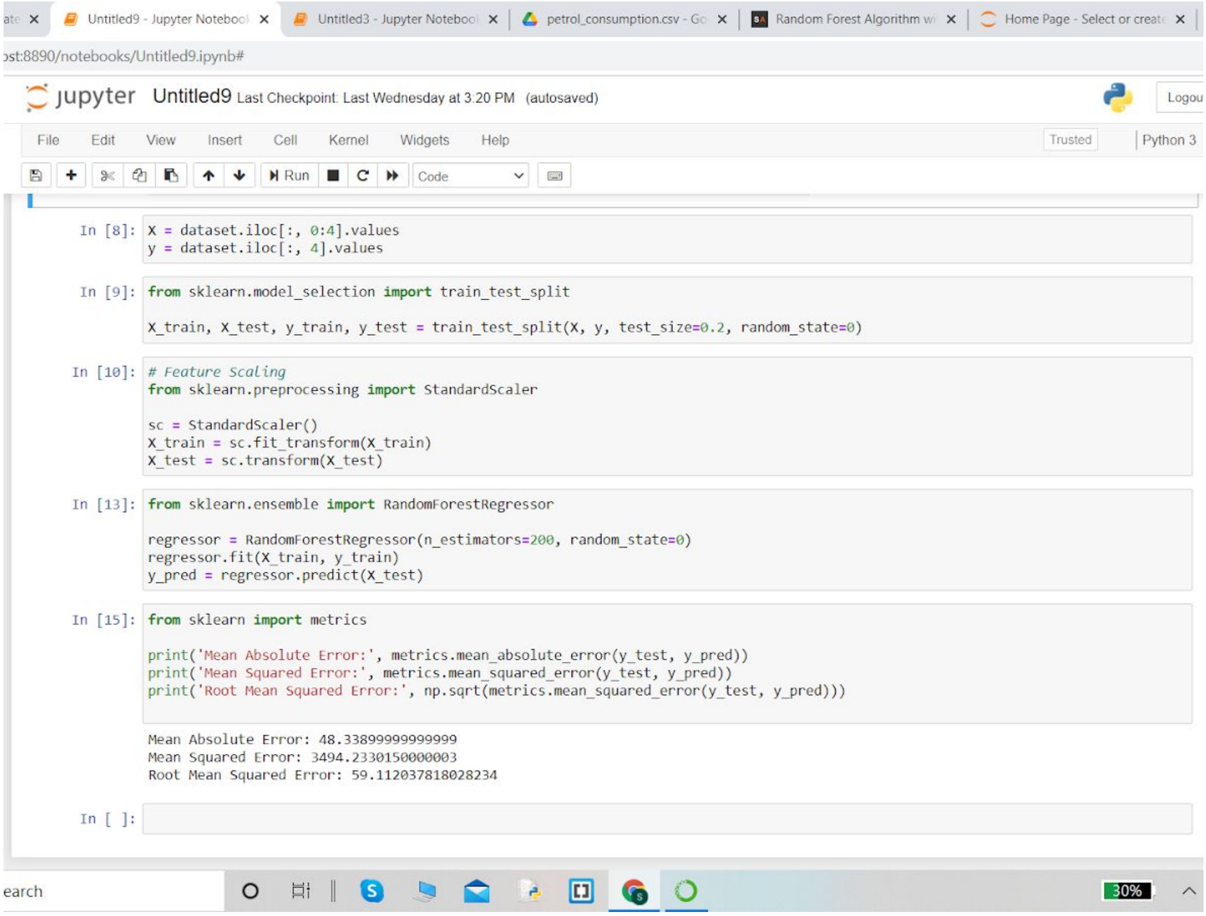
	Petrol_tax	Average_income	Paved_Highways	Population_Driver_licence(%)	Petrol_Consumption
0	9.0	3571	1976	0.525	541
1	9.0	4092	1250	0.572	524
2	9.0	3865	1586	0.580	561
3	7.5	4870	2351	0.529	414
4	8.0	4399	431	0.544	410

```
In [7]: dataset.tail()
```

Out[7]:

	Petrol_tax	Average_income	Paved_Highways	Population_Driver_licence(%)	Petrol_Consumption
43	7.0	3745	2611	0.508	591
44	6.0	5215	2302	0.672	782
45	9.0	4476	3942	0.571	510
46	7.0	4296	4083	0.623	610
47	7.0	5002	9794	0.593	524

```
In [8]: X = dataset.iloc[:, 0:4].values
        y = dataset.iloc[:, 4].values
```


Untitled9 - Jupyter Noteboo
Untitled3 - Jupyter Noteboo
petrol_consumption.csv - Go
Random Forest Algorithm wi
Home Page - Select or create
ost:8890/notebooks/Untitled9.ipynb#
jupyter
Untitled9 Last Checkpoint: Last Wednesday at 3:20 PM (autosaved)
Logou
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3
Run
Code
earch
30%

```
In [8]:  X = dataset.iloc[:, 0:4].values
         y = dataset.iloc[:, 4].values

In [9]:  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=0)

In [10]: # Feature Scaling
         from sklearn.preprocessing import StandardScaler

         sc = StandardScaler()
         X_train = sc.fit_transform(X_train)
         X_test = sc.transform(X_test)

In [13]: from sklearn.ensemble import RandomForestRegressor

         regressor = RandomForestRegressor(n_estimators=200, random_state=0)
         regressor.fit(X_train, y_train)
         y_pred = regressor.predict(X_test)

In [15]: from sklearn import metrics

         print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred))
         print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_pred))
         print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))

         Mean Absolute Error: 48.33899999999999
         Mean Squared Error: 3494.2330150000003
         Root Mean Squared Error: 59.112037818028234

In [ ]:
```

```
import pandas as pd
import numpy as np

dataset = pd.read_csv(r"Downloads\petrol_consumption.csv")

dataset

dataset.head()

X = dataset.iloc[:, 0:4].values
y = dataset.iloc[:, 4].values

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=0)

# Feature Scaling
```

```
from sklearn.preprocessing import StandardScaler

sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

from sklearn.ensemble import RandomForestRegressor

regressor = RandomForestRegressor(n_estimators=200, random_state=0)
regressor.fit(X_train, y_train)
y_pred = regressor.predict(X_test)

from sklearn import metrics

print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_pred))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
```

basic steps involved in performing the random forest algorithm:

1. **Pick N random records from the dataset.**
2. **Build a decision tree based on these N records.**
3. **Choose the number of trees you want in your algorithm and repeat steps 1 and 2.**
4. **In case of a regression problem, for a new record, each tree in the forest predicts a value for Y (output). The final value can be calculated by taking the average of all the values predicted by all the trees in forest. Or, in case of a classification problem, each tree in the forest predicts the category to which the new record belongs. Finally, the new record is assigned to the category that wins the majority vote**