Software Engineering Tools Lab

Assignment No-2

(Module 2- Software Development Frameworks)

Name – Pratik Mukharu Raut

PRN – 2019BTECS00050

Batch – T7

Name – Kshitija Jadhav

PRN – 2019BTECS00053

Batch – T7

Software – Node.js

1. **Original Author :** Microsoft Corporation, Ryan Dahl, OpenJS Foundation, Bryan Cantrill

2. **Developers :** OpenJS foundation

3. **Initial release :** May 27, 2009; 12 years ago

4. **Stable release :** (16.14.0)

5. **Preview release :** 19-10-2021

6. **Repository (with cloud support) :**
https://github.com/nodejs/node

https://cloud.google.com/nodejs

7. **Written in (Languages) :** JavaScript, C, C++, CoffeeScript

8. **Operating System support :** z/OS, Linux, macOS, Microsoft Windows, SmartOS, FreeBSD, OpenBSD, IBM AIX

9. Platform ,portability : Node JS runs on V8 engine

10. Available in (Total languages) :

11. List of languages supported : JavaScript is the only language that Node.js supports natively, but many compile-to-JS languages are available.

12. Type (Programming tool, environment etc.) : runtime environment

13. Website : https://nodejs.org/en/

14. Features :

1. Commendable data processing ability
2. Active open-source community
3. Additional functionality of NPM
4. Advanced hosting ability of NodeJs
5. Fast data streaming
6. Cross-platform compatibility
7. The convenience of using one coding language
8. V8 Engine
9. Facilitates quick deployment and microservice development
10. Scalable

15. Size (in MB, GB etc.) : 436 MB

17. Type of software (Open source/License) : Open source

18. If License- Provide details : No License

19. Latest version : 17.5.0

20. Cloud support (Yes/No) : YES (Google Cloud)

21. Applicability:

High performance and scalability, Freedom to develop cross platform applications

22. Drawbacks:

- One of the biggest disadvantages of Node. js is that it is unable to perform heavy computation tasks.
- Node. js is capable of executing JavaScript code on a single thread basis

Q2 Implement linear regression problem using Google colab (Perform preprocessing, training and testing)

Google Collab Implementation - https://colab.research.google.com/drive/179QRfTav0Xwp9Ksqi0dUWdJkqHzCr5Fs#scrollTo=X7oCmYGwvpkW

PDFCopy of Google Collab –

```
#NAME - Aditya Hemant Sarnobat
#PRN - 2019BTECS00042
#BATCH - T4

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import preprocessing, model_selection, metrics
import warnings
warnings.filterwarnings("ignore")

data = pd.read_csv('/energydata_complete.csv')
data.head()
```

	date	Appliances	lights	T1	RH_1	T2	RH_2	T3	RH_3	
0	2016-01-11 17:00:00	60	30	19.89	47.596667	19.2	44.790000	19.79	44.730000	19
1	2016-01-11 17:10:00	60	30	19.89	46.693333	19.2	44.722500	19.79	44.790000	19
2	2016-01-11 17:20:00	50	30	19.89	46.300000	19.2	44.626667	19.79	44.933333	18
3	2016-01-11 17:30:00	50	40	19.89	46.066667	19.2	44.590000	19.79	45.000000	18
4	2016-01-11 17:40:00	60	40	19.89	46.333333	19.2	44.530000	19.79	45.000000	18

```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
from sklearn.model_selection import train_test_split
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression

# 75% of the data is usedfor the training of the models and the rest is used for testing
train, test = train_test_split(data,test_size=0.25,random_state=40)
train.describe()
```

	Appliances	lights	T1	RH_1	T2	R
count	14801.000000	14801.000000	14801.000000	14801.000000	14801.000000	14801.000
mean	97.835281	3.809202	21.685153	40.271333	20.343487	40.432
std	102.928289	7.940816	1.605537	3.983201	2.199037	4.081
min	10.000000	0.000000	16.790000	27.233333	16.100000	20.463
25%	50.000000	0.000000	20.745000	37.363333	18.790000	37.900
50%	60.000000	0.000000	21.600000	39.656667	20.000000	40.500
75%	100.000000	0.000000	22.600000	43.090000	21.533333	43.290
max	1080.000000	60.000000	26.260000	63.360000	29.856667	56.026

```
# Divide the columns based on type for clear column management

col_temp = ["T1"]

col_hum = ["RH_1"]

col_weather = ["T_out", "Tdewpoint","RH_out","Press_mm_hg",
                    "Windspeed","Visibility"]
col_light = ["lights"]

col_randoms = ["rv1", "rv2"]

col_target = ["Appliances"]

train_X = train[col_temp]
train_y = train[col_hum]

test_X = test[col_temp]
test_y = test[col_hum]

print(train_X)
print(train_y)
```

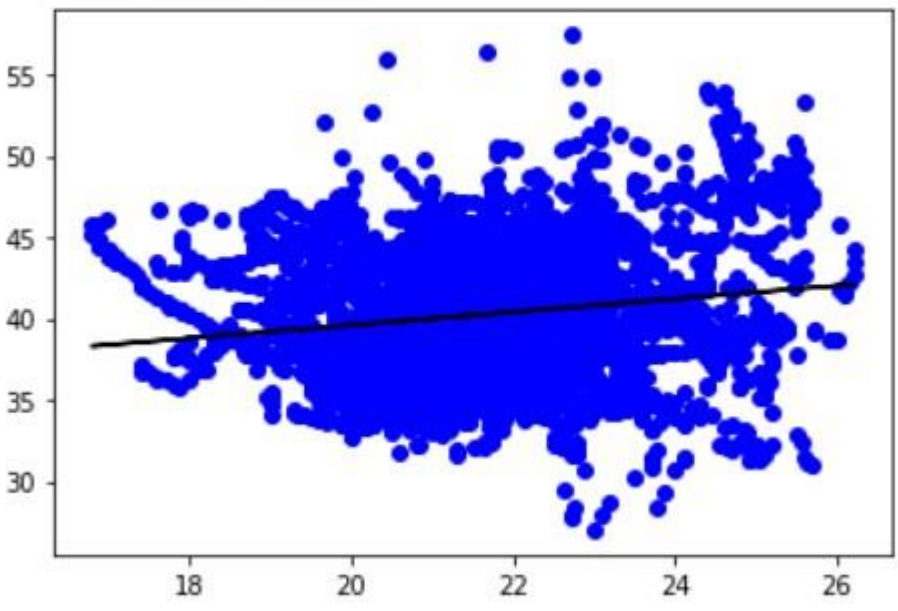
```
      T1
9544  22.600
19366 23.700
10816 22.200
15457 21.290
2956  21.290
...    ...
11532 21.760
16065 21.200
14501 21.600
14555 23.175
11590 21.600

[14801 rows x 1 columns]
          RH_1
9544  34.700000
```

```
19366  40.290000
10816  45.800000
15457  35.790000
2956  47.400000
...         ...
11532  41.933333
16065  36.433333
14501  37.466667
14555  37.722500
11590  38.700000

[14801 rows x 1 columns]
```

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression


regr = LinearRegression()
regr.fit(train_X,  train_y)

print(regr.score(test_X,  test_y))

pred_y = regr.predict(test_X)
plt.scatter(test_X, test_y, color ='b')
plt.plot(test_X, pred_y, color ='k')

plt.show()
```

```
0.34809281277703413
```

```
0.028200704451733838
```



1s co