**SET Lab**

**Assignment No.: 02**

**Om Khairnar (2019BTECS00078)**

**Yash B. Pandhare (2019BTECS00085)**

**Batch: T4**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Q. 1. For NodeJS answer the following questions.**

**NodeJS FRAMEWORK**

1. **Original author -** Node.js was written initially by **Ryan Dahl** in 2009, about thirteen years after the introduction of the first server-side JavaScript environment, Netscape's LiveWire Pro Web. The initial release supported only Linux and Mac OS X. Its development and maintenance was led by Dahl and later sponsored by Joyent.
2. **Developers -** OpenJS Foundation
3. **Initial release -** May 27, 2009
4. **Stable release -** . 17.4.0 / January 18, 2022
5. **Preview release-** .NET 5.0
6. **Repository (with cloud support ) -** [github.com/nodejs/node](file:///E:\TY\SET%20Lab\assignment2\github.com\nodejs\node)
7. **Written in (Languages)- C, C++, JavaScript**

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

1. **Platform ,portability -** Visual Studio
2. **Available in (Total languages) -** JavaScript is the only language that Node.js supports natively, but many compile-to-JS languages are available
3. **List of languages supported -**. JavaScript is the only language that Node.js supports natively, but many compile-to-JS languages are available. As a result, Node.js applications can be written in CoffeeScript, Dart, TypeScript, ClojureScript and others. Node.js is primarily used to build network programs such as Web servers.
4. **Type (Programming tool, integrated development environment etc.)** Runtime environment
5. **Website -** [nodejs.org](https://nodejs.org/)
6. **Features-**
   * + 1. [Node.js](https://www.geeksforgeeks.org/nodejs-tutorials/)is a cross-platform runtime environment that allows you to create server-side and networking applications. Node.js apps are written in JavaScript and run on OS X, Microsoft Windows, and Linux utilizing the Node.js runtime.
       2. Node.js also comes with a large library of JavaScript modules, making it much easier to construct web applications with it. It enhances the functionalities of Node.js.
       3. NodeJs facilitates the integration of programming languages with APIs, other languages, and a variety of third-party libraries. It is used exclusively in the ‘JavaScript everywhere’ paradigm for web app development and can handle both server-side scripting and client-side programming.
       4. Web development is a constantly evolving process that demands ongoing innovation and updates to keep up with the desire for game-changing technologies as each year passes.
       5. Most developers favor JavaScript for front-end programming, which has recently been boosted by NodeJS for back-end development. NodeJS is being used in mobile application development in addition to online development.

NodeJs = Runtime Environment + Javascript library

* + - 1. Object Oriented: A huge complaint against NodeJS was down to its JavaScript heritage, which frequently involved lots of procedural spaghetti code. Frameworks like CoffeeScript and TypeScript solved these issues, but came as a bolt on for those who seriously cared about coding standards. Now with the release and general adoption of ES6, Classes are built into the framework and the code looks syntactically similar to C# , Java and SWIFT.

### OVER 600,000 FREE OPEN SOURCE PACKAGES ON NPM: The Node community is enormous and the number of permissive open source projects available to help you save time is mind boggling. These libraries range from simple helpers and charts to full blown frameworks.

1. **Size (in MB, GB etc.)-** 26.7 MB
2. **Type of software (Open source/License) -** Node.js is an **open-source, cross-platform, back-end JavaScript runtime environment** that runs on the V8 engine and executes JavaScript code outside a web browser.
3. **If License Provide details.:** MIT License
4. **Latest version:** node-v16.14.0 0-x64.msi
5. **Cloud support (Yes/No)-** Yes, Node. js on Google Cloud **integrates with Cloud Monitoring, Cloud Trace, Cloud Logging, and Error Reporting**, allowing you to transparently instrument live production applications to diagnose performance bottlenecks and software bugs.
6. **Applicability-** The popular Uses of Node.js are as below:
   * + Browser Games
     + Chat Rooms
     + Collecting data: Massive amounts of data can be collected and made more efficient with [the uses of Node.js](https://www.educba.com/events-in-node-js/). large quantities of data into a database usually create a bottleneck, because database access is a blocked operation. Node.js, on the other hand, receive this data, then send it to the backend in a piecewise manner.
     + Streaming
     + Real time Applications:  Example: – Trello, DropBox.
     + API on top of Object DB.
7. **Drawbacks (if any) -**

* Reduces performance when handling Heavy Computing Tasks.
* Node.js invites a lot of code changes due to Unstable API.
* Node.js Asynchronous Programming Model makes it difficult to maintain code.
* Choose Wisely – Lack of Library Support can Endanger your Code.

**Q.2.** **Implement linear regression problem using Google colab (Perform preprocessing, training and testing).**

Dataset- <https://archive.ics.uci.edu/ml/datasets/Water+Quality+Prediction>

Link for google collab: [https://colab.research.google.com/drive/1mqdEIycTquiRRLyKsmM-lprGWxZnPsZ9?usp=sharin g](https://colab.research.google.com/drive/1mqdEIycTquiRRLyKsmM-lprGWxZnPsZ9?usp=sharing)

Code -

|  |
| --- |
| ## water quatity prediction  import matplotlib.pyplot as plt |
| import numpy as np |
| from sklearn import linear\_model |
| import pandas as pd |
| from sklearn.metrics import mean\_squared\_error, r2\_score |
| from sklearn.model\_selection import train\_test\_split |
| from google.colab import files |
| uploaded = files.upload() |
| df = pd.read\_csv('AirQualityUCI.csv') |
| df.head() |
| x=df.iloc[:,2].values |
| y=df.iloc[:,3].values |
|  |
| x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = 1/3, |
| random\_state = 0) |
|  |
| x\_test=x\_test.reshape(-1,1) |
| x\_train=x\_train.reshape(-1,1) |

|  |
| --- |
|  |
| lin\_reg=linear\_model.LinearRegression() |
|  |
| lin\_reg.fit(x\_train,y\_train) |
|  |
| lin\_reg\_pred=lin\_reg.predict(x\_test) |
|  |
| print("Coefficients:\n",lin\_reg.coef\_) |
| print("Intercept:\n",lin\_reg.intercept\_) |
|  |
| print("Mean squared error: %.2f" |
| % mean\_squared\_error(y\_test, lin\_reg\_pred)) |
|  |
| # variance prediction |
| print('Variance score: %.2f' % r2\_score(y\_test, lin\_reg\_pred)) |
|  |
| plt.scatter(x\_test, y\_test, color = 'red') |
| plt.plot(x\_test, lin\_reg\_pred, color = 'blue') |
| plt.title('Temperature vs Humidity(Test set)') |
| plt.xlabel('Temperature') |
| plt.ylabel('Relative Humidity') |
| plt.show() |

Result:

