**Software Engineering Tools Lab**

**Assignment No-2**

**(Module 2- Software Development Frameworks)**

**Batch – T8**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Name** | **PRN** |
| 1 | Sayali Yogesh Desai | 2020BTECS00206 |
| 2 | Tejal Satish Pansare | 2020BTECS00203 |

1. List of Frameworks/IDEs/Software’s

**a. Eclipse**

**b. Android SDK**

**c. Node.js**

**d. DotNet**

**e. Ruby on Rails**

**f. Anaconda**

**g. Google Colab**

For every Frameworks/IDEs/Software’s given above provide the answers for below questions

We are choosing **Node.js** as Frameworks/IDEs/Software’s for below questions

|  |  |  |
| --- | --- | --- |
| 1 | Original author | Ryan Dhal |
| 2 | Developers | Open JS Foundation |
| 3 | Initial release | May 27, 2009; 12 years ago |
| 4 | Stable release | 17.4.0/ January 18, 2022; 21 days ago |
| 5 | Preview release | 0.10.42/ February 2016 |
| 6 | Repository (with cloud support) | <https://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 |
| 9 | Platform, portability | Cross-Platform, iTwin.js |
| 10 | Available in (Total languages) | 1 |
| 11 | List of languages supported | JavaScript (CoffeeScript, Dart, TypeScript, ClojureScript and others) |
| 12 | Type  (Programming tool, integrated development environment etc.) | Runtime Environment |
| 13 | Website | <https://nodejs.org/> |
| 14 | Features | Single Threaded, Asynchronous, Event Driven, Open Source, Fast Performance, Highly Scalable, No Buffering, Caching, Licensed |
| 15 | Size (in MB, GB etc.) | By default, Node.js (up to 11. x) uses a maximum heap size of 700MB and 1400MB on 32-bit and 64-bit platforms, respectively. |
| 16 | Privacy and Security | NPM Phishing,  Regular expressions Denial of Service (DOS) |
| 17 | Type of software  (Open source/License) | License |
| 18 | If License- Provide details | MIT License |
| 19 | Latest version | Node v17.4.0 |
| 20 | Cloud support (Yes/No) | Yes |
| 21 | Applicability | * Internet of Things * Real-Time Chats * Complex Single-Page Applications * Real-Time Collaboration Tools * Streaming apps * Microservices Architecture |
| 22 | Drawbacks (if any) | * Inability to process CPU bound * Cell back hell issue * Application Programming Interface is not scalable * Performance bottlenecks with heavy computation |

**1. Implement linear regression problem using Google colab**

**(Perform preprocessing, training and testing)**

Dataset 1- <https://www.kaggle.com/spittman1248/cdc-data-nutrition-physical-activity-obesity>

Dataset 2- <https://archive.ics.uci.edu/ml/datasets/Air+Quality>

Dataset 3- <https://archive.ics.uci.edu/ml/datasets/Appliances+energy+prediction>

Dataset 4- <https://archive.ics.uci.edu/ml/datasets/Bike+Sharing+Dataset>

Dataset 5- <https://archive.ics.uci.edu/ml/datasets/Demand+Forecasting+for+a+store>

Dataset 6- <https://archive.ics.uci.edu/ml/datasets/Hungarian+Chickenpox+Cases>

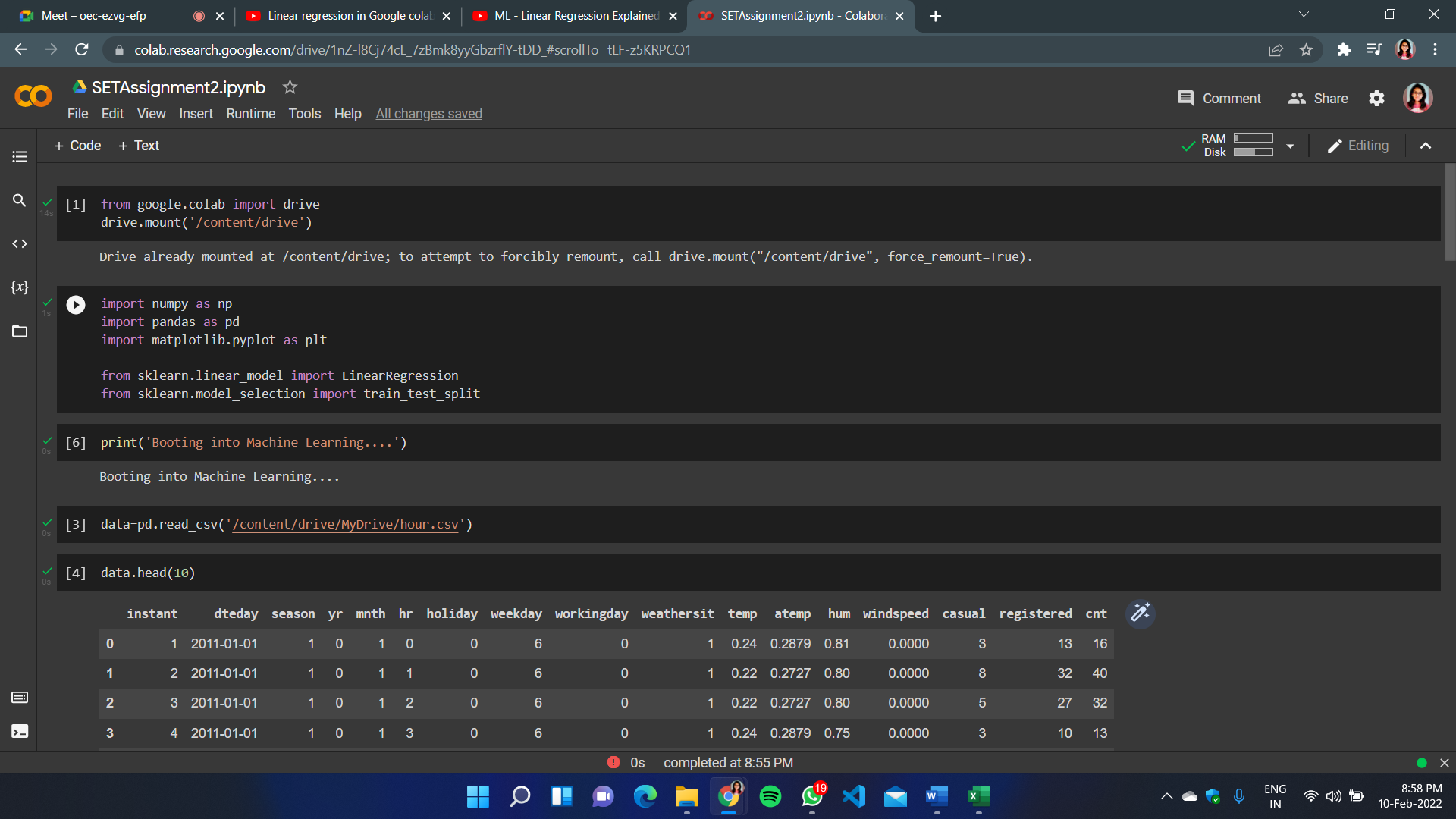
Dataset 7- <https://archive.ics.uci.edu/ml/datasets/KDD+Cup+1998+Data>

Dataset 8- <https://archive.ics.uci.edu/ml/datasets/Water+Quality+Prediction>

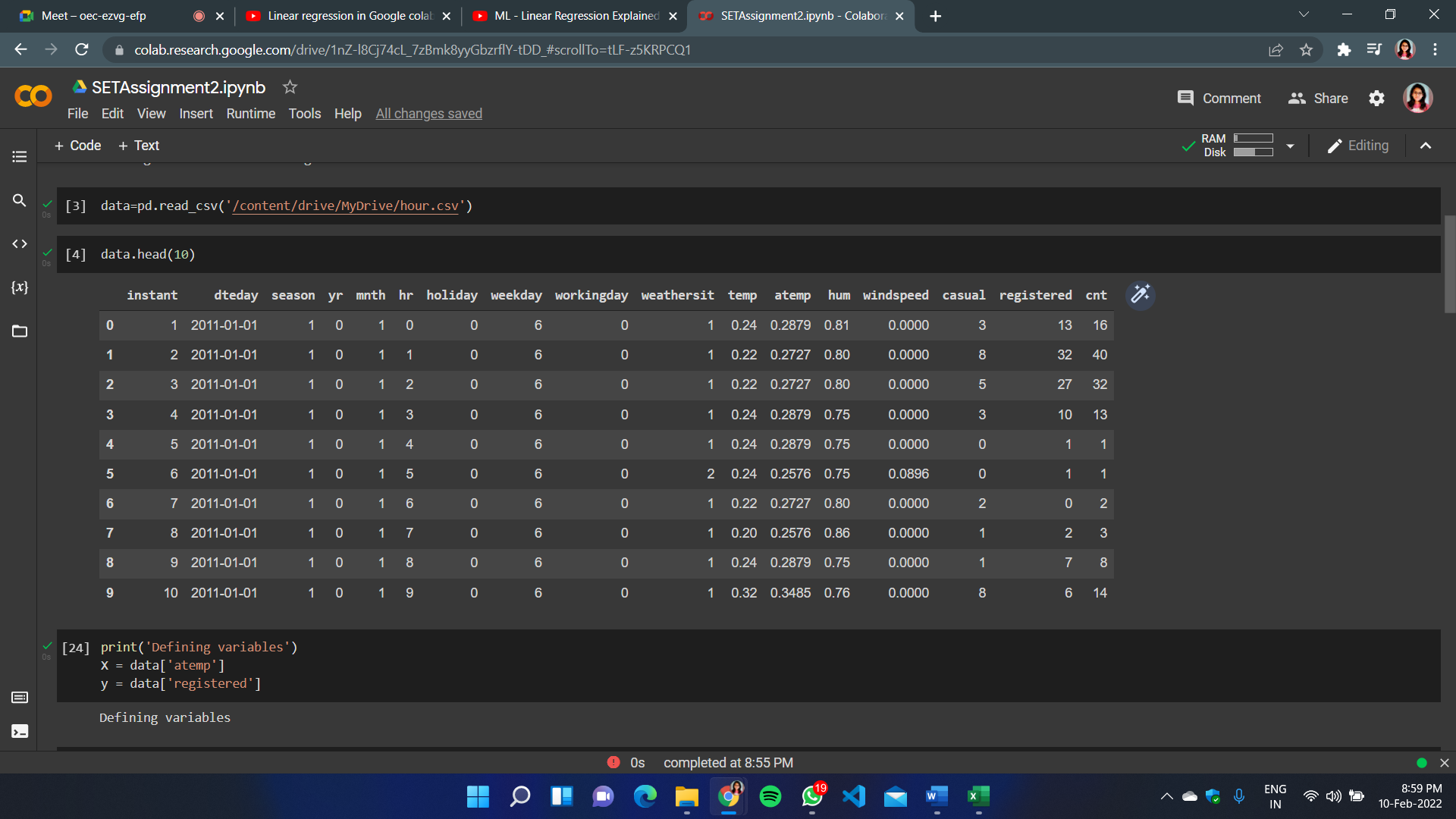
We have used Dataset no. 4 i.e. <https://archive.ics.uci.edu/ml/datasets/Bike+Sharing+Dataset> for implementation of linear regression problem using Google colab.

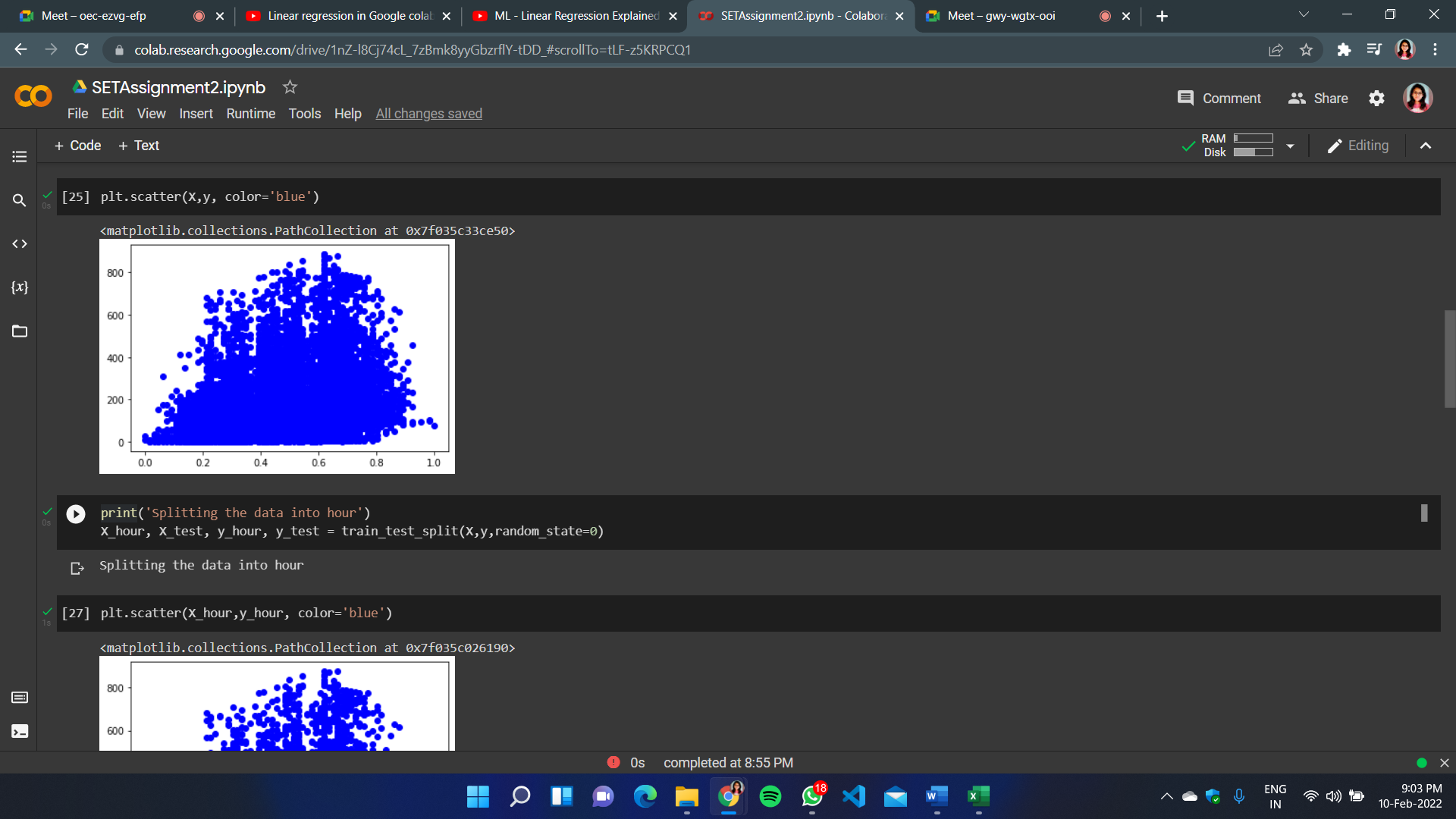
My Performance: <https://colab.research.google.com/drive/1nZ-l8Cj74cL_7zBmk8yyGbzrflY-tDD_?usp=sharing>

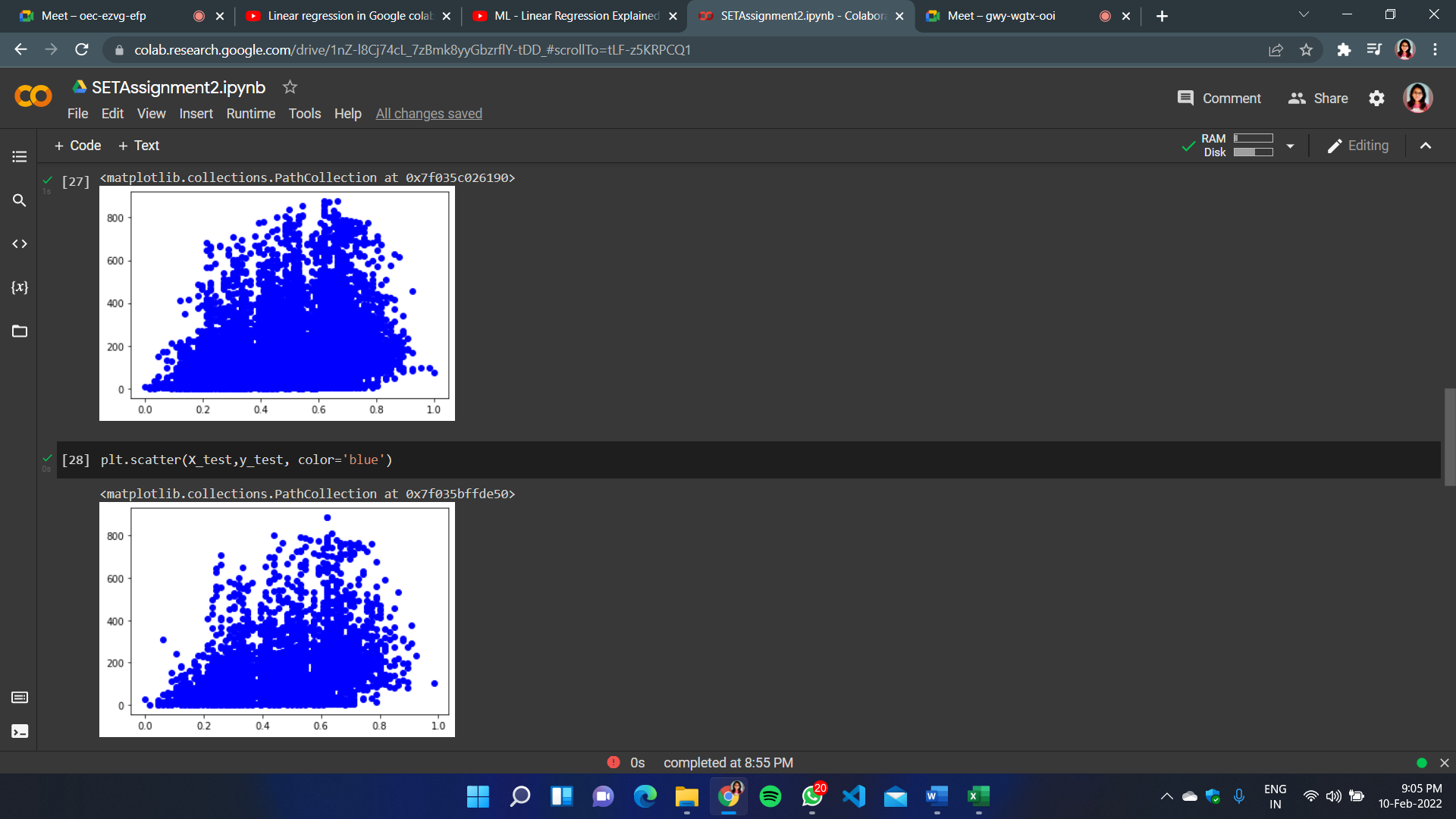
* **Preprocessing:**



* **Training:**







* **Testing:**

